

DOCUMENT RESUME

ED 284 539

IR 012 782

AUTHOR Wilson, Thomas C., Ed.
TITLE MACUL '87 Proceedings. Selected Papers. Annual Conference of the Michigan Association for Computer Users in Learning (Detroit, Michigan, March 26-27, 1987).
INSTITUTION Michigan Association for Computer Users in Learning, Ann Arbor.
PUB DATE Mar 87
NOTE 65p.
PUB TYPE Collected Works - Conference Proceedings (021) -- Viewpoints (120) -- Reports - Descriptive (141)
EDRS PRICE MF01/PC03 Plus Postage.
DESCRIPTORS *Computer Literacy; *Computer Software; *Computer Uses in Education; Curriculum Development; Elementary Secondary Education; Higher Education; Interactive Video; Latin; Programing
IDENTIFIERS LOGO Programing Language; Spreadsheets

ABSTRACT

Selected from presentations made at the 1987 conference of the Michigan Association for Computer Users in Learning (MACUL), the 13 papers in this proceedings address topics related to computer applications, computer literacy, and software applications: (1) "Keys to Programming Success for College Students" (Henry S. Austin); (2) "Electronic Spreadsheet Applications for Classroom Administration" (Gerald L. Beebe); (3) "Toward the Compleat Latin Computer Classroom" (Robert L. Black); (4) "Computer Curriculum Guide: A Developmental Process" (Gloria J. Earl); (5) "The Computer Literacy Debate" (Paul Kofi Egbo and Carol E. Bacek); (6) "Logo in the Outside World" (Mark Guzdia); (7) "Video Job Center" (Ray Hoag); (8) "Computer to Computer Communication" (Richard G. Lewandowski); (9) "Computers and the Arts" (Scott Michael Lis); (10) "An Educator's View of Software Piracy" (Sharon Octernaud); (11) "Michigan Occupational Data Analysis System--Assessment Systems" (Chris M. Olson and Carol E. Culpepper); (12) "Data Management to the Aid of Discovery" (Mike Rains); and (13) "Bibbits, Gribbits and Other Logical Things for Grades 1-3" (Charlotte Scherer). (MES)

* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *

ED284539

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

☒ This document has been reproduced as
received from the person or organization
originating it.

☐ Minor changes have been made to improve
reproduction quality.

• Points of view or opinions stated in this docu-
ment do not necessarily represent official
OERI position or policy.

MACUL

Michigan Association for Computer Users in Learning

Proceedings SELECTED PAPERS

Annual Conference
Detroit, Michigan

March 26 - 27, 1987

Thomas C. Wilson, Editor

"PERMISSION TO REPRODUCE THIS
MATERIAL HAS BEEN GRANTED BY

Thomas C. Wilson

BEST COPY AVAILABLE

IR012782

© 1987 by Michigan Association for Computer Users in Learning

No permission is necessary to reproduce articles from the **MACUL '87 Proceedings** for non-commercial use provided appropriate reference is made to the **MACUL '87 Proceedings** and to the author.

No part of this book may be reproduced, stored in a retrieval system, or transcribed, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, for commercial use without the prior written permission of the **MACUL** Board.

Disclaimer: Authors were requested to send camera-ready copy. All selected materials have been duplicated "as-is." **MACUL** and its designates assume no responsibility for grammatical errors or poor image quality. Opinions expressed herein do not necessarily reflect the opinions of **MACUL** or its affiliates.

With great pleasure I present the **MACUL '87 Proceedings**. Many aspects of educational computing are represented. Surprisingly, we did not receive any duplicate subjects. Included are items addressing software applications in unique arenas, new perspectives on programming, and a fresh look at literacy, to name a few.

The growth of educational computing across the country has been phenomenal. Decreasing prices and increasing awareness renew the spirits of those who have been around awhile and captivate the new-comers. One of the most remarkable occurrences, however, has been the endless creativity of educators in developing new applications for both the old favorites and the latest products.

It is this creativity, with its diversity in experience levels, machine preferences, and software attractions, that gives us, at once, our strength and our challenge. This year's **Proceedings** reflects the richness of **MACUL** with research findings, personal scenarios, and controversial viewpoints. A taste of the broader wealth of experience can be found here.

Selection was based on clarity of presentation, organization of ideas, and adherence to accepted semantic and syntactical standards. I trust you will find the contents instructional, innovative, and inviting.

I would like to thank Christine Canning and Elizabeth Carlson for their assistance in successfully producing the **MACUL '87 Proceedings**. Their insight has indeed been helpful. In addition, Bruce Maxim, Richard Moll, Pamela Moore, Richard Parres, Frank Schemanski, and Deborah Young served as reader-evaluators. For their willingness to participate, I am grateful.

Thomas C. Wilson
Ann Arbor
March 1987

Contents

1. Keys to Programming Success for College Students Henry S. Austin, Ph.D.	1
2. Electronic Spreadsheet Applications for Classroom Administration Gerald L. Beebe	6
3. Toward the Complete Latin Computer Classroom Robert L. Black	11
4. Computer Curriculum Guide: A Developmental Process Gloria J. Earl, Ph.D.	15
5. The Computer Literacy Debate Paul Kofi Egbo and Carol E. Bacek	19
6. Logo in the Outside World Mark Guzdial	24
7. Video Job Center Ray Hoag	29
8. Computer to Computer Communication Richard G. Lewandowski	33
9. Computers and the Arts Scott Michael Lis	36
10. An Educator's View of Software Piracy Sharon Octernaud	41
11. Michigan Occupational Data Analysis System - Assessment Systems Chris M. Olson and Carol E. Culpepper	46
12. Data Management to the Aid of Discovery Mike Rains	51
13. Bibbits, Gribbits and Other Logical Things for Grades 1 - 3 Charlotte Scherer, Ph.D.	56

Keys to Programming Success for College Students

Dr. Henry S. Austin PhD CDP
Professor of Computer Science
Oakland CC - Orchard Ridge Campus
Farmington Hills, Michigan 48018
313-471-7726

Introduction

The keys to success in the first programming course (FPC) in college are **ability, attitude, awareness and participation**. The bases for this assertion are the findings from both a review of the literature on computer programming achievement and a research study on the same topic (Austin). The article will begin with a brief statement on the problem of the low rate of successful completion in the FPC, followed by background discussions on the literature findings, the research study and the findings from the research study. The article will conclude with an examination of the four keys to FPC success.

Problem

Only about half of the students enrolled in the first programming course at the college level complete the course with a grade of 'C' or better (ACE, Lee). This high rate of unsuccessful completion is detrimental to the self-esteem of both instructors and unsuccessful students and also wastes educational resources since the FPC is over-subscribed at most schools.

Strategies to raise the successful completion rate without diluting course quality involve either selecting FPC enrollees on the basis of a high success probability or modifying course treatment for unsuccessful students. Both strategies are dependent on the identification of the student characteristics which are linked to successful course completion.

Previous Research Findings

The literatures on both general freshman achievement and computer programming achievement (CPA) were extensively reviewed, leading to the following observations:

- 1) The four major categories for classifying student characteristics were demographic, academic, ability and personality.

- 2) Almost all of the studies focused on freshman students in four year institutions; the few studies on community college (CC) students were over ten years old.
- 3) Previous academic achievement and both quantitative ability and background were the leading predictors of CPA for freshman students in four year institutions; demographic and personality characteristics as well as prior programming background were either insignificant or of minor importance.
- 4) The few studies on CC student CPA suggested that general ability was the leading CPA predictor for CC students.
- 5) Program reading tests, program writing tests and program writing lab exercises were identified as CPA measures. Almost all of the CPA studies used a composite measure of course performance without delineating the component measures.

Research Study

The author investigated the relationships between student characteristics and CPA measures for a population of CC students because of both the problem of unsuccessful FPC completion and the relative lack of current studies in this area on CC students. The research model is displayed in figure one.

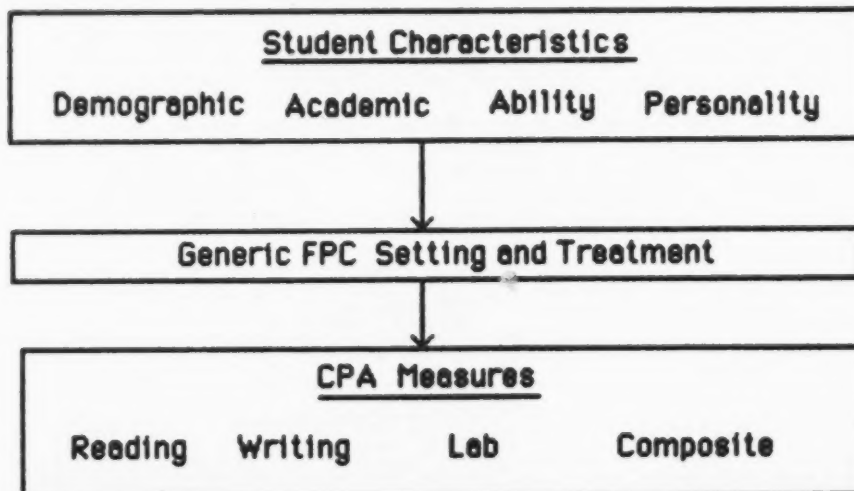


Figure 1 -- Research Model

The population, setting and treatment were 162 students enrolled in the Fall, 1984 day sections of the FPC at Oakland CC with generic lecture/lab/discussion treatment. A variety of questionnaires and psychological tests were used to measure the four categories of student characteristics. Test and exercise results from the FPC were used to measure the CPA categories. The characteristic measures were completed by 98 of the 162 students and of those 98, 76 also completed all of the CPA measures. A variety of multivariate statistical techniques were used to analyze the relationships of student characteristics to CPA.

Research Study Findings

- 1) The most important predictors for each of The CPA components were:

Reading CPA:

Quantitative and algorithmic abilities and high school grade average.

Writing CPA:

Verbal ability and the absence of negative computer attitudes.

Lab CPA:

Age, personality and gender.

Composite CPA:

Composite ability and the absence of negative computer attitudes.

- 2) The composite predictor model was almost 90% accurate in predicting student placement into succeed/fail completion categories based on student characteristics.
- 3) Introvert/analytic personality characteristics were linked to high CPA while extrovert/synthetic characteristics were linked to low CPA.
- 4) Negative computer attitudes were linked to low CPA while the relationships of positive computer attitudes to CPA were insignificant.
- 5) Self-assessed math ability was significantly related to CPA.
- 6) Previous computer coursework and computer interest measures were both unrelated to CPA.

Keys to College Programming Success

Ability. Both the literature review and the research study findings emphasize the linkage between high CPA and both quantitative and pattern detection abilities. The research study also links high CPA with verbal ability for CC students. These findings attest to the primacy of basic skills to the successful completion of the FPC in college.

Attitude. The research study results suggested that there was no relationship between positive attitudes towards computers and programming achievement but that a negative attitude towards computers was related to low achievement. It should be emphasized that this relationship was statistically independent from ability influences on achievement. Students should be aware that a negative attitude towards computing is self-defeating; what is linked with successful achievement is the absence of a negative attitude.

Awareness. The findings from both the literature review and the research study point to the somewhat paradoxical conclusion that neither interest in computing nor self-assessed programming ability nor programming background were linked to successful achievement in the college FPC. Therefore, students should not enter the college FPC with a false sense of optimism if they possess any of these characteristics but instead be aware that the course will challenge them regardless of their computer interest and experience.

These findings also point to the need for increased communication between the secondary and college levels in order to develop a mutual understanding of the computer course requirements at both levels.

Participation. The research study findings support a linkage between low achievement and failure to do programming lab exercises, particularly for both younger and male students. The self-discipline to participate in the programming exercise component of the FPC outside of scheduled class time is a requirement for course success.

Concluding Remarks

Successful completion of the college FPC is primarily dependent on a combination of adequate quantitative and verbal skills, absence of negative attitudes towards computers and programming lab participation outside of scheduled class time. There is no research support for any relationship between either computer interest or programming background or self-assessed programming ability and CPA.

A variety of strategies seem necessary in order to raise the level of successful completion in the college FPC. Secondary and college instructors need to communicate their course objectives to each other. Students need to be counseled on the ability and attitude prerequisites of the FPC as well as the task requirements of the FPC and the importance of lab participation. Finally, there is research support for a less rigorous FPC as an alternative for students lacking the ability or need for the rigor of the regular FPC (Campbell). Such a course could serve as both a prerequisite to the regular FPC for slower students as well a service course for students in other disciplines.

Additional research is needed to clarify the relationships between student characteristics and different measures of programming achievement in the FPC at both the community college and senior college levels.

References

ACE 1984: Association of Computer Educators. Annual meeting of an informal consortium of Michigan computer educators.

Austin, H. 1986: Associations of student characteristics to measures of computer programming achievement for suburban community college students. dissertation: University of Michigan. pub. University Microfilms International: Zeeb road, Ann Arbor, Mi. order# 8612466.

Campbell, P. 1984: The effect of a preliminary course on performance in a traditional programming course. ACM SIGCSE bulletin: V16 # 5.

Lee, J. 1985: Careers in computing. Communications of the ACM: V28 #5.

Author's note: A more detailed summary of the research study is contained in the February, 1987 proceedings of ACM-SIGCSE.

Electronic Spreadsheet Applications for Classroom Administration

Gerald L. Beebe
Vocational Education Department Chairperson
Hartland Consolidated Schools
9525 Highland Road
Hartland, MI 48029
(313) 632-6363

The use of electronic spreadsheets can greatly facilitate the many daily tasks of both the classroom teacher and administrator. Tasks such as grading, fundraising, budget management, and materials inventory can be accurately and quickly recorded on these spreadsheets as well as conveniently saved for future reference.

Spreadsheet templates can be purchased through many commercial firms, through computer-user groups such as the National Appleworks User Group (NAUG), or, as in the author's case, developed yourself to fit personal needs. The advantage of buying a prepared template is saving time. The advantage of doing it yourself is designing a template to meet your needs within your system rather than making your system fit someone else's ideas.

Ideas presented in this paper are not meant to be complete nor all-encompassing. They are presented to the reader to spark interest and perhaps give the reader new ideas for spreadsheet applications. The ideas can be utilized with most spreadsheet programs including VisiCalc, Lotus 1-2-3, and Appleworks as well as many more.

Gradebooks

Perhaps the most obvious use of electronic spreadsheets is for the classroom gradebook. There are many fine gradebook programs commercially available. However in most cases the teachers will have to redesign their grading systems to "fit" the template.

As stated before, developing your own system lets you keep the system you may have developed over the years and transfers it to the computer. Although initially time consuming when entering names and

setting up columns, once developed the spreadsheet becomes a convenient and useful tool for determining student grades anytime during the semester. (These lists of names can be transferred to other spreadsheets for such things as fundraisers, milk money, and attendance.) In our school system, teachers must determine grades at the end of each marking period and semester, during the middle of each marking period for progress reports, and weekly for athletes to determine athletic eligibility. As long as the spreadsheet is kept up-to-date, it is a simple matter of pressing a few keys to determine exactly the academic status of the student. This is also a useful tool if the student, parent, counselor, etc. needs to know the present grade status for a student.

At parent-teacher conferences it is also possible to give a parent a detailed, typed report of their students progress. This report could also be sent home at any time during the year. Detailed reports often create a much greater impact than a note saying the student isn't doing to well.

The following example of a gradebook was done on Appleworks as were all of the examples in this report. Many of the templates for these examples are available through NAUG. (NOTE: When printed as a spreadsheet rather than on a word processor, "wrap-around" would not occur as in this example. Also many of the following examples are partial therefore the totals may be inaccurate.)

GRADEBOOK 1986-87																									
Marketing 3RD Hour 1st Marking Period																									
Quizzes T											Homework														
Reaction Papers											Major														
Scores																									
Names		1 2 3 4 5 6 7 8					1 2 3 4 5 6 7					T	1 2 3 4 5 6					T01	T02						
PstrEssayRePrT03 Qz		Hwk RP TOTALS %																							
Crashcup, Clyde		5 6 10 9 8 5 43 0 1 1 1 1 1					5	15 20 20 19 17 91					80	92											
44 50 90 92 71		83.3 91 494 86.75%																							
Dingle, Joe		1 3 0 2 5 2 1 14 1 1 1 1 1 1					6	17 20 19 18 15 89					56	63											
38 0 87 54 23		100 89 510 63.79%																							

In this example up to six quizzes, seven homework assignments, five written papers, three tests plus three projects are entered, averaged, and totaled. Some spreadsheet programs have applications that will

allow you to include a letter grade look-up, class averages, and other features you may desire for your gradebook. This same class list could be transferred to other spreadsheets for things such as attendance and participation points.

Fundraising

Keeping track of fundraising statistics is one of the headaches many educators face as part of their year's schedule. Many fundraising companies provide bookkeeping services while others provide minimal assistance. By using a self-designed spreadsheet, accurate tallies of sales can be easily kept and adapted to whatever needs desired.

In the following partial spreadsheet, the student's name and number of each type of product sold is entered. The computer then figures out the subtotal value for each item for each student as well as the totals for the group. Similar spreadsheets could be used for such record keeping chores as lunch money, dues, and other student income and expenditures.

1986 DECA FUNDRAISER														
BENSEN'S														
NAME	COOKIES			FRUIT CAKE			OTHER			TOTALS				
	CC	OR	PS	30	20	10	PIN	STK	AMB	CCC	COOKIES	FRUITCAKE	OTHER	TOTAL DUE RE
BROWNLEE	4					2		5		2	\$14.00	\$7.00	\$30.00	\$51.00
BAKE	33	9	10	4	5	19	3	3	1	4	\$122.00	\$126.50	\$50.50	\$361.00
				2				2						
TOTALS	170	21	23	5	21	34	20	23	8	24	\$749.00	\$265.00	\$340.00	\$1,431.00 \$1

Income Statements

If an operation such as a student store or student organization exists, it may be necessary to provide an income statement such as the following:

DECA SALES REPORT			
		OCTOBER, 1986	
		INCOME	DISBURSEMENTS
BEGINNING BALANCE	\$1,123.52	\$70.80	\$51.08
OCTOBER INCOME FOR CANDY,		\$166.91	\$260.00
FLOWERS, SLUSH, DUES	\$2,859.85	\$60.95	\$48.40
OCTOBER DISBURSEMENTS	\$2,420.63	\$94.21	\$46.30
ENDING OCTOBER BALANCE	\$1,562.74	\$202.65	\$87.50
		\$2,859.85	\$2,420.63

In this simple statement the beginning balance is entered along with all income and expenditures for the time period involved. The computer then totals both income and disbursement columns and resolves an ending balance. Obviously a much more complex system and statement could be designed to handle other types of similar functions.

Letters

Although not a regular practise for the electronic spreadsheet, there are times when a letter may be written on the sheet instead of transferring the data to a word processor. Short messages can easily be written in the cells of the spreadsheet as in the case of the following order for supplies.

Pacific Balloon Mfg. Co. Inc.

Please send us the following items:

1 F02/SN	\$10.20
1 F03/RPHV	\$10.56

TOTAL ENCLOSED WITH ORDER	
TO ESTABLISH CREDIT	\$236.23

SINCERELY,

This example is part of a lengthy order. The computer increases the accuracy in the resulting total. If a lot of text is desired, transferring the spreadsheet to a word processor may be preferred since using the spreadsheet for this purpose may become cumbersome and time consuming when printing the final results.

Budgeting

One of the most efficient administrative uses of the spreadsheet is for budgeting purposes. The following partial spreadsheet might be used by a department head or vocational director. Updated weekly, this budget gives the administrator an immediate, up-to-date picture of any budgeted area.

In this example an expenditure is entered in the appropriate column such as Instructional Supplies. The computer totals the column and subtracts it from the amount budgeted. This information is also useful when determining new budgets or honoring purchase requests.

HARTLAND HIGH SCHOOL
VOCATIONAL EDUCATION
BUDGETS 1986-87

AUTOMOTIVE-VESTON

Account Number	Description	85-86	86-87	Total
11-127-				Left
3220-030-532	InService Training	\$75.00	\$150.00	\$137.00
4120-030-532	Equipment Maintain	\$1,000.00	\$1,500.00	\$165.00
5154-030-532	Instr. Supplies	\$2,615.00	\$6,500.00	\$2,472.17
6400-030-532	Equip/Furn New	\$1,000.00	\$2,000.00	\$1,832.30
	Totals	\$4,690.00	\$10,150.00	\$4,626.47

Instr. Travel	InService Train	Instr. Supplies	Equip. New
11/11 \$45.60	11/11 \$13.00	7/11 \$90.74	9/18 \$67.00
		7/11 \$1,839.06	9/18 \$80.70
TOTALS	\$13.00	\$4,027.83	\$147.70

Summary

The value of electronic spreadsheets can be summarized as follows: 1) they make it easy to store and retrieve data in a small amount of space, 2) data can be compiled and calculated quickly and accurately, and 3) data can be organized to fulfill personal needs to make one's job easier. If the spreadsheet design used doesn't meet these values then either design or buy a new one or use something other than an electronic spreadsheet such as a word processor, data base, or pen and paper.

Spreadsheet use is limited only by the imagination, restrictions of the software, and the amount of K in the computer's memory. For example, this author uses spreadsheets for a variety of reports for a school store including a merchandise perpetual inventory. Through practice and experimentation, the user can quickly become an experienced applicant and developer of electronic spreadsheets.

Toward the Compleat Latin Computer Classroom

Robert L. Black
Foreign Language Department
Wyoming Park High School
2125 Wrenwood S.W.
Wyoming, MI 49509

I started out as a Latin teacher. Oh, I had looked at computers as early as 1969, taking a few classes in Fortran and program design as a graduate student at the University of Minnesota. But the computer consisted of three floors of vacuum tubes surrounding one terminal, and seemed as far removed from practical classroom use as typewriters from Horace Mann.

Computers had already made inroads into schools by the late 70's, when I was told by a textbook author that micros could not be used for foreign language instruction since it took almost their entire memory to get them to display accent marks, not to mention exotic alphabets such as Greek. Dr. Sweet, who had written a programmed learning course in Latin, had investigated the computers of the day as a possible format for his work. It finally appeared as 'Artes Latinae', a programmed instruction textbook.

My first serious consideration of the microcomputer as a practical tool for the average classroom teacher came in 1981, when a student whose father was a programmer brought in a xerox of a magazine article. It described a program designed to conjugate Latin verbs. It seems the same factors which make computers so logical for use with math make them very adaptable to highly inflected languages with rigid syntax. The program really didn't accomplish anything more spectacular than the Cuthbertson Verb Wheel, which had been designed in the 1930's. Both reminded me of the fifth century B.C. Greek mechanical computer found some years ago on a Roman ship, lost at sea as some Neronian Goering was having it abducted from Athens' Tower of the Winds.

But the idea had intrigued me, so I latched on to a couple of Apple II's which, having been forced on some reluctant science teachers, were partaking in some dust gathering experiments. Thus began the search for programs. A few hangman games notwithstanding, there was not much of use in the 5,000 (count them) programs said to be available. Oh, there were word processors, crossword puzzle generators, and test creators. The MECC series was useful in that regard. But they all suffered from the same deficiency, from the practical teacher's viewpoint. All were incompatible with each

other. I.E., material typed in with great care for one program's use was not consistent with the format of the next program, and had to be completely reentered.

Enter the parent who had restarted me on my computer quest. He volunteered to write a program which would do most of what I had in mind, enabling students to play 'games' on the computer which in fact drilled them on the same files which would later be used to create their tests. His programs worked nicely, although he complained somewhat about the lack of programming features on the Apple, compared to the TRS and other machines.

At the same time, I tried to obtain more computers for my classroom. But there was money for computers in science or math classes, not in subjects such as Latin. A sympathetic administrator offered matching funds if Latin Club could raise the balance. We looked at prices. Three to five Commodore 64's, a newly released computer, could be purchased for the price of one Apple. I asked my programmer, who had been joined by another volunteer, an IBM programmer by profession. Both were impressed with the 64's superiority in programmability, graphics, sound, and price. Many of my students had 64's of their own, outnumbering Apples by more than 20:1. An added advantage would accrue from students being able to work at home. We ended up with eight 64's.

In the beginning, all classroom work was done on the programs written by the two volunteers, ten programs in all. All were copyrighted and issued to other teachers as shareware, with contributions being made commensurate with their usefulness.

At first, using computers was more work than not doing so, but after the first few months of typing all test and study materials into a master database, the effort began to pay off. The first indication was in the Latin verb system, always a burden for students to master. Whereas formerly they would know noun forms adequately, I concentrated drill and 'game' materials on the verb system, using the games my volunteer parents had written as well as commercial programs for which they wrote conversion files, such as Davidson's 'Word Attack'.

Lo and behold! The noun system, without computer assistance, became the comparatively weaker form set. The computer had helped them conquer three moods, two voices, six tenses, and dozens of other odds and ends like conjugations, periphrastics, and ablative absolutes.

I was sold on computers. Gradually, every facet of classroom procedure, including roll taking, lesson plans, seating charts, grading, overhead transparencies, reports home, classroom papers, and the school paper (produced on GEOS), was done with

computers. Since I only teach three Latin classes, I began to branch into other subjects: American literature, mythology, journalism, world literature, English grammar.

A typical day in the life of one of my (Latin) students might go like this: Roll is taken from a seating chart printed by computer. Space is provided on this chart for daily oral grades. The computer reassigns seats every few weeks based on these grades, assuring that students who need help are seated in front. (He who is nearer the shepherd is better fed.) An assigned student loads 'Mnemosyne', a programmed learning 'game' into the computers. Eight students quietly take their places at computers and spend 15 minutes working through modular explanations of whatever grammatical concept is currently being taught: e.g. the gerundive of necessity.

The remainder of the class works on Latin readings from their textbooks (done as a homework assignment) in a very traditional manner, translating in turn as I call on them. When one group finishes at the computer, 'Mnemosyne' will save their record to disk, and another group takes over at the computers. Near hour's end, a one sentence quiz, based on the reading of the day, or a grammatical form, might be given (produced from the unit's database).

Another day they might use 'The Choice of Hercules', 'Hemlock', or 'Socratic Quiz', to practice topics previously covered in traditional textbook fashion. 'Word Attack' and 'Spell it!', by Davidson, and 'Dialog', by Codewriter, are also used occasionally. Some days we play a bingo game produced from the one time typed in database, or a crossword puzzle or word find based on Latin vocabulary.

At unit's end, the database may churn out a review sheet, or at any rate a test in any one of the standard types. The multiple choice test printer is capable of printing out tests numbering hundreds of questions. We have never been able to reach its capacity, which is limited only by the memory of the particular Commodore computer being used. It will even make up the wrong answers for you, if you wish, and independently randomly generates both the questions and the answers from the database you choose. I can print an individualized multi page test for each student in less time than I could formerly type one ditto.

And mistakes are fewer. Spelling and typos can be easily corrected and poorly worded questions can be reworked when noticed, or removed from the database. Many time consuming repetitious tasks can now be relegated to the machine and my time spent on the creative part of teaching.

Since the vast majority of students who have computers of their own have Commodore 64's, many of my students use the same programs at home. The advanced ones even create files for the database, while others turn in their homework on disk or printout. Many use a program called 'Dictionary' while doing their translations. Instead of looking the word up the old fashioned way, they type in the first few letters and an asterisk, and the computer looks up the meaning! A few students have even shown up with Vic 20's, so we modified some programs for that machine, along with some versions for Pets which some schools have requested. Until the multitude of compilers came along, we had the program check at the beginning to see which machine it was in, and modify itself accordingly.

The accent marks? Oh yes, the 64's can do accent marks, and even Greek, both on and off the screen. Using the Fontmaster series of word processors, by Xetec, our programs can print in Chinese, Arabic, and Hebrew, the latter two typed in from right to left, of course.

It is of course, a never ending story. New uses keep cropping up, new programs beg to be written. Half a dozen are on our drawing boards now. But I like to think that, if Socrates ever visited the Tower of the Winds, marvelling at the workings of the world's first computer, he would certainly have approved of our methods of guiding the minds of young and old to knowledge. One program may lead to another, but so too does one question.

Computer Curriculum Guide : A Developmental Process

Dr. Gloria J. Earl
Computer Coordinator
Warsaw Community Schools
P.O. Box 288
Warsaw, IN 46580-0288
(219) 267-3238

At a recent conference which I attended on Interactive Instruction, I heard some outstanding leaders in the field of computer education make what to some might be considered startling comments. Harvey Long, Educational Consultant for IBM began by asking: "Is there a compelling reason to use technology in the schools?" (Long, 1986). He went on to stress that if technology is to be used it must be relevant to the faculty, students and world of work (Long, 1986). Tom Synder, software developer for Synder Productions stated that he was suspicious that software didn't teach (Snyder, 1986). He believes that if computers are to be used they must relate to how teachers relate to students. Computers don't belong in the classroom if we can't find the appropriate software to meet our purposes. Snyder firmly believes the teacher is the key in this search (Snyder, 1986).

While these comments might be considered radical by some, I found them to be a breath of fresh air among the cries of those who state unequivocally that the computer is the answer to all of our educational woes. They summarize a philosophy which underlined a project our school district had of developing a Computer Curriculum Resource Guide. From initial idea to the final printing and distribution we firmly believed that if the computers are to be effectively used by teachers, they must see the relevancy to their classroom and students. They must be shown how the computer can assist them rather than replace them. Teachers must see the interrelationship between technology and the curriculum and be the final decision maker in the selection of software. The purpose of the guide was to assist teachers in seeing these connections and making these decisions.

The process of development of the resource guide became as important as the actual product in meeting this goal. It would have been relatively easy to simply produce a computer curriculum guide sitting in the coordinator's office invisioning what we would like to have happen with computers in the school. The committee felt strongly that if this was done, as so often curriculum guides are written, that the teachers would add it to the dusty collection on their selves and would not look at it again. We believed that it was essential to keep in mind the end user as the guide was developed if we wanted the guides to become a true resource for the teachers. In an effort to do this, various levels of teacher input was sought throughout the process.

The first step in developing the resource guide was to determine what we wanted the guide to accomplish and what needed to be included in the guide in order to accomplish it. This first step was taken by the district-wide computer committee which had representation from each of the schools in the district.

The purpose of the guide was to define a scope and sequence of computer skills k-12 and provide teachers with resources for teaching these skills. In addition the guide was designed to illustrate a tie of the computer to the present curriculum. It was believed that the majority of the computer skills contained in the scope and sequence could be taught by using the computer as a learning tool.

In order to accomplish this purpose, the committee felt that the guide should contain the following materials: a scope and sequence of computer skills; classroom activities, resources and materials to teach the skills; a listing of the library software; all forms and procedures presently used in the program (i.e. repair, ordering, etc.); specific skills list for each subject in each grade level; a list of available district software cross-referenced with the skills list; and a teacher comment section.

Secondly, the committee took a look at what teacher input should occur. They felt that it was important that response was received from both computer users and novice. If we were designing the guide for all teachers, we needed the non-user to evaluate its usefulness for them as one of the end users. Committee also felt that it was important to have input from all grade levels and all curriculum areas.

The third important task for the committee was to look at a format that they felt would be best received by teachers. We believed that we must stay flexible and open to suggestions throughout the process. While we needed a starting place, we had to realize that if the process was important the final format would be much different.

Once the purpose was established, the type of input was determined and the original format was decided, we were ready to select a core group of people to work with the computer committee on the project. This selection took on several forms all of which had volunteerism as a key element. We solicited volunteers through an open letter to all teachers describing the project and giving them the opportunity to offer various levels of expertise and time commitment. We also requested principals to encourage people whom they felt would be an asset to the project. Others were approached by computer committee members. The teachers could volunteer to be writers, evaluators or curriculum sources. The majority of the final group were gotten through the open letter.

We were now ready to begin the process of writing. As a starting point the committee tackled the job of developing a scope and sequence of computer literacy skills and related classroom activities. Input was gathered by them on an individual basis in the buildings.

Parallel to the development of the scope and sequence was the development of skills list for each area of the curriculum. Since our goal was to integrate the computer into the present curriculum rather than creating a new curriculum, the list of skills taught at each grade level for each subject area was requested from the entire teaching staff through the principals. A composite of these list was then developed by the coordinator. The volunteer group was then given the list to modify and develop a consistent format for each grade level. The final composite in the consistent format was then sent out to teachers for response.

The next process was to begin to identify the software that was being utilized in each grade level to either introduce, reinforce, remediate, or enrich each of the skills identified on the skills list. Again all teachers were requested to provide input through the principals. They were asked to list only those programs that they were presently using. The teacher input list was then compiled. Following a training which was aimed at expanding the ways that we view how software can be utilized, the core committee was charged with compiling a total list cross-referencing all software owned by the district with the curriculum skills. This process was the most lengthy and, in the minds of the committee, the most important. Many catalogs are beginning to come out with similar ideas of the various things that can be taught with a specific piece of software. We felt that often these suggestions were unrealistic from a real educational point of view. We, therefore, insisted that someone in the corporation must have used or evaluated the programs before they were listed. A small group from the core committee was appointed to complete this process during the summer. The process was considered important enough that these teachers were paid a stipend for eight days of summer work to complete the task.

The responsibility for the final format and additional materials also fell to this group. The summer group was made up of a teacher from each grade level so that questions concerning appropriateness of skills or software could be answered from someone who deals daily with students.

The last step of the developmental process was to send out a draft to the core committee for input into the final format and content. Suggested changes were then made by the coordinator and taken to the original computer committee for a final overview and a discussion as to distribution. It was felt that it would be necessary to give the guides to the teachers face to face with an explanation as to their content and potential usefulness.

The distribution was done on a one-to-one basis with the computer coordinator meeting with each teacher for a twenty minute conference to go over the guide and respond to any questions regarding the guide or other computer concerns. This also gave the coordinator the opportunity to suggest that teachers keep their guides upgraded as software was ordered, utilize the teacher comment sheets for planning, and identify software needs based on skills taught.

The response to the guide was overwhelmingly positive. It helped teachers to see that there was a compelling reason to use computers in the school. It helped teachers to see the relationship of the technology to the classroom educational goals, and assisted them in seeing that they, the classroom teacher, is the most important decision maker in the process of integrating computers into the curriculum. It is our belief that the process of development was as essential as the end product in meeting these goals.

REFERENCES

Long, Harvey. Consortium for Interactive Technology. Annual Conference. Norfolk, Virginia. November, 1986.

Snyder, Thomas. Consortium for Interactive Technology. Annual Conference. Norfolk, Virginia. November, 1986.

The Computer Literacy Debate

**Paul Kofi Egbo and Carol E. Bacak
AFAM International Educational Consultancy
723 S. Francis Avenue
Lansing, MI 48912
(517) 372-2741 and (313) 334-4877**

The computer revolution has come along with its own jargon often referred to as computerese. Existing words, concepts and expressions have acquired new meanings and significance in the computer community. One such computerese which is of particular interest to educators is the concept of computer literacy. Although the concept has become fashionable and ubiquitous in computer literature, it has defied a common definition acceptable to all, and is generating heated debates among computer technologists, educators and the general public.

The two extreme views of the computer literacy controversy may be summarized as follows: The "Comprehensive" school of thought led by scholars such as Dwyer and Critchfield (1981), Bork (1982), and Anderson (1983) holds that a computer literate person should know everything about computers from their history, technical components, how computers operate, their uses and social implications, ethics, how to use computers to perform given tasks, to how to program computers to perform those tasks. The "Tool" school of thought maintains that the computer is just another tool devised by humans to help them perform certain tasks. Consequently, a user need not have detailed technical knowledge about computers or how to program them. What is critical is for the individual to use the computer to meet his or her own needs. Megarry (1983) argues the logic of this viewpoint thus:

...It would be wholly unjustifiable to restrict driving licences to such would-be drivers as can explain the principles of the four-stroke combustion cycle, and just as irrational to prevent students from discussing the environmental impact of motorized transport unless they could first pass a Heavy Goods Vehicle Proficiency test (p. 20).

One reason for the uncompromising feud over what is, or what is not, computer literacy is that for some computer experts, the word "literacy" takes on a specialized, almost-sacrosanct, lexical significance when married to the magic word "computer." Most dictionary definitions of "literacy" incorporate the notion of the ability to read and write. By extension, a functionally literate person is a person who has the ability to read and write to enable him or her to function in society. If this basic premise is accepted, then it follows that there cannot be one literacy level for everybody. For example, it would be idiotically farcical to expect a factory assembly line worker to have the same level of reading and writing as a lawyer!

It is also curious that the invention of other technologies such as the typewriter or television were not accompanied by definitive requirements for a person to become "typewriter-literate" or "television-literate." Does a classroom teacher using television for instruction need to understand what a cathode-ray tube is or be able to explain sound modulation in order to use the medium effectively? However, the same logic fails with some experts when applied to computer literacy.

The diversity of views on computer literacy was amply demonstrated at a conference of computer scientists, teachers, producers of educational materials, publishers, vendors and government officials held in Reston, Virginia in 1980 and sponsored by the National Science Foundation. The aim of the conference was to seek goals for achieving national computer literacy in the United States by 1985. After two and a half days of deliberations, perhaps the most gratifying conclusion of the experts was "The recognition that the concept of computer literacy is multi-faceted" (Deringer and Molnar, 1982, p. 5). In other words, computer literacy means different things to different people.

The argument for the "tool" view of computers is strengthened by the fact that unlike in the early days of computers when a few specialists were compelled to learn about and to use computers, today "We don't use the computer; we make use of it through its applications" (Zamora, 1983, p.8). It is also true that particularly since the arrival of the micorcomputer

which made possible the proliferation of computers, there has been a growing market for software, a need which is being met by commercial companies and publishing houses which have been producing software for business, industrial, administrative, home, and educational applications. Thus, the individual does not need to produce his or her own program but can use or adapt an existing piece of software.

In K-12 education, the early curricula for computer instruction were mainly designed for junior and senior high schools and tended to follow the "comprehensive" view. There was very little, if any, stress on software utilization (see Computer Literacy and Awareness for Students (CLAS), Pilot Edition, by the Tri-County Computer Consortium, Nov. 1981). The introduction of instructional computing at the K-6 level has added to the problem of what is to be taught and at what level.

The problem with the "comprehensive" approach to computer literacy with its variety of components such as history, ethics, understanding of hardware, and programming, is that it invokes in teachers what we call the "Add-on Effect;" teachers see instructional computing as another subject area they are required to cover.

It is our view that any computer literacy program must come to grips with the reality that an individual's role places different demands on his or her use of, and involvement with computers. As Seidel (1982) succinctly put it:

...An administrator in a school, or a policymaker, must be able to appreciate how computers can be used to, for example, augment the existing disciplines or how they might aid in managing student schedules. However, it is debatable whether such individuals need to become computer programmers in order to attain this appreciation. My guess is that for now the latter is not the case (pp. 23-24).

Computer technology and the industry have developed to the level where our concern should be on how to use the computer as a tool and where everybody

should not be expected to know everything about computers. While a programming language such as logo should be encouraged, particularly at the lower grades, because of its ability to build up interest in younger children, as well as its value for exploring geometry and for problem solving, it is doubtful that dexterity in a language like BASIC or Pascal is necessary for the majority of K-12 students. Apart from logo, programming should be reserved for those students at the secondary level who are headed for specialized careers such as programmers or computer analysts. Emphasis for the generality of students should be on software utilization and applications such as word processing and database usage.

To minimize the add-on effect, the goal of education should be to integrate computer instruction into the existing curriculum. For example, units on history and ethics would be better taught as part of social studies where concepts such as the development of technology and issues relating to the uses of technology should be part of the curriculum. Word processing can be a part of the language curriculum. Database usage can be integrated into several curricular areas particularly science and social studies.

Historically, emphasis in technology has been more on the ability to use a tool rather than a profound knowledge of its history, technical details, and how it works. The computer should be no exception.

References

- Anderson, Ronald E. "National Computer Literacy, 1980." In Computer Literacy, pp. 9-17. Edited by Robert J. Seidel, Ronald E. Anderson and Beverly Hunter. New York: Academic Press, 1982.
- Bork, Alfred. "Computer Literacy for Teachers." Ibid., pp. 91-98
- Deringer, Dorothy K. and Molnar, Andrew R. "Key Components for a National Literacy Program." Ibid., pp. 3-7.

Dwyer, T.A. and Critchfield, M. "Multi-computer Systems for the Support of Innovative Learning." In Computer Assisted Learning, pp. 7-12. Edited by P.R. Smith. Oxford: Pergamon, 1982.

Megarry, Jacquetta. "Thinking, Learning, and Educating: the Role of the Computer." World Yearbook of Education 1982/83: Computers and Education. Edited by Jacquetta Megarry, David R.F. Walker, Stanley Nisbet and Eric Hoyle. New York: Nichols Publishing Company, 1983.

Seidel, Robert J. "On the Development of an Information Handling Curriculum: Computer Literacy, A Dynamic Concept." In Computer Literacy, pp. 19-32.

Zamora, Ramon. "Computers and Other Literacies." In Computers in Curriculum and Instruction, pp. 6-11. Edited by Tim Grady and June D. Gawranski. ASCD, 1983.

Logo in the Outside World

Mark Gusdial
Bell Communications Research
Room 1H-218
331 Newman Springs Road
Red Bank, NJ 07701-5699
(201) 758-2381

Introduction

Logo has been used for educating youth since its creation (Papert, 1980). Its user-modifiable and understandable language, friendly turtle graphics, and powerful programming constructs have made it a tremendous tool for exploring ideas with a computer.

The same benefits that make Logo useful for children also make it a useful tool in worlds outside of the explicitly educational environment. It seems reasonable to expect that a package that is understandable to children would also be understandable to the (hopefully) more sophisticated adults. The power of its graphic and programming constructs extend from the applications of children to the applications of grown-up children.

Even when it is adults using the language and not children, Logo is still a tool for education. Logo is useful in educating children because it allows them to explore worlds that might otherwise be closed or difficult for them. Similarly, Logo can allow adults to explore ideas that might otherwise be hard to access for them. A prime characteristic in several of the examples below is that the programs are designed to allow the user to explore microworlds using Logo.

In this paper, I will present some Logo programs that have been developed for outside world situations. After the discussion of these programs, I will suggest why Logo was useful in these situations.

Examples

Two of the example Logo programs that I will be describing are programs that have been written to solve actual problems. The latter two programs have been written to solve hypothetical problems, similar to problems encountered in the outside world.

Example 1 - The Vector Microworld

The first example comes from a problem that I faced in my work at Bell Communications Research (Bellcore). I am a Member of Technical Staff (MTS) at Bellcore in a center involved in statistical research. One of my recent assignments was to write a program for a microcomputer based on a complex statistical algorithm currently running on a large minicomputer that used a sophisticated vector-oriented programming language, AT&T Bell Laboratories' S (Becker, 1985). S is a language whose primitive operations involve things like matrix multiplication, rotating matrices, and performing vector addition in a single operation. This means that not only was the algorithm that I was translating complex, but the language of the algorithm was far advanced over the languages I would normally use on the microcomputer (e.g., C or Pascal).

I decided that my first priority was to get the algorithm running on the microcomputer in a language familiar to me so that I might explore the algorithm

and then make my necessary improvements and modifications to meet my purposes.

I chose to write a vector-oriented microworld in Logo to serve as my host language on the microcomputer. I simulated the advanced minicomputer language in Logo by writing each of the sophisticated operations as a Logo function and naming the functions like the S operations. In this way, I could make a mechanical, line-by-line translation from the original Bellcore program written in S into a new program written in my Logo microworld without really understanding the function of the whole routine beforehand. Then, once I had it in running in a familiar environment (a friendly statistician provided me with test values to verify my translation), I could play with the code until I understood how the algorithm of the original program worked.

The microworld was developed in a single afternoon, and has been an effective tool for exploring the algorithm.

Example 2 - Data to Tones Translation

Mark Wruble is a PhD student in Psychology at the University of Florida. His research requires him to gather reams of discrete data and then analyse it for variations in cycles (variations that may or may not appear in any set of data) - a tedious and time-consuming process.

An idea occurred to Mark on how he might be able to analyze the data in the same way without the tedium. The human ear is very good at detecting changes in cycles. If the data values could be mapped to tones and played as music, it might be easier to detect variations by listening to the data. Mark wanted to use his ears rather than his eyes for detecting variations.

Since this mapping idea was a sideline to his real research, a sideline that might not even prove useful, few resources could be allocated to the project. An extensive programming effort with a long development time was out of the question. An IBM PC was the only available computing system.

To meet the obligations described above, the data-to-tones mapping package, called DATATONE, was written in IBM PC Logo. Developing in Logo permitted experimentation with the mapping procedures after only a couple hours of development. Further, IBM PC Logo has a standardized tone generating primitive so no unusual features had to be implemented.

The environment of Logo helped with the user interface as well. DATATONE requires the dataset to reside in a disk file for processing. **EDITFILE** provides a simple editor for entering datasets with some 4000 to 8000 elements. Logo variables can be used to hold lists of data values for passing to the mapping procedures as arguments.

Three separate sets of procedures were included in the DATATONE package. The first takes a dataset file and displays each unique data element in the file and the number of occurrences of that element in the file. This helps the user in determining what element should correspond to which tone. The second two are variations in the mapping and playing procedures; one which assumes the same duration for each note, and another which allows each unique data element to have its own duration.

There are no results to report yet from the use of this microworld, but it has provided a testbed at little or no cost for the experiment.

Example 3 - The CHEM Little Language

In his October 1986 column in *Communications of the ACM*, Jon Bentley (1986) wrote on Little Languages. Bentley says in that article that "A computer language enables a textual description of an object to be processed by a computer program" and that "a little language is specialized to particular problem domain and does not include many features found in conventional languages." Further, a little language should imply that the user can make use of it in perhaps a half hour, master it in a day, and the development should not take more than a few days.

One of Bentley's examples was a little language that he and his colleague, Brian Kernighan, wrote for typesetting chemical formulae, called CHEM. CHEM allowed the user to specify a chemical formulae in chemical language, and then it typesets the described formulae.

Bentley and Kernighan's little languages struck me as being similar to Logo microworlds (Papert, 1980). As an exploration of little languages as microworlds, I decided to attempt implementing a turtle graphics version of CHEM in Logo. My goal was to create a Logo microworld where the description of a chemical formulae in order to draw its diagram could be specified in chemical language.

The Logo CHEM is turtle-oriented, but it understands phrases in bond and atomic notation. The various CHEM commands cause the diagrams to appear at the position of the turtle in the direction of its current heading. These commands are used both to draw the formulae and to position the turtle, such as C, H, CH₂, BENZENE, and O for atomic symbols and combinations, and BOND, and DOUBLEBOND for drawing bonds and positioning. For example, methane can be drawn with the Logo CHEM sequence `LEFTSTART H BOND CH2 BOND H`.

The Logo CHEM microworld could not be used for the same purpose as the little language CHEM, unless some ambitious user worked at connecting a typesetter to their Logo-running microcomputer. It could be used to produce high quality chemical drawings by hooking up a graphics printer or plotter in a standard manner. Regardless of its use for driving output, the main goal of Logo CHEM was to produce the drawings of the little language CHEM using chemical language, thus demonstrating Logo's use in creating a language and environment for a particular problem domain.

Example 4 - The Dynamic Map

I recently read about the controversy between various Ancient History and Literature scholars over the position and existence of the city of Troy which fell to Greek soldiers by way of the Trojan Horse in Homer's *Odyssey*. Most of the arguments seemed to be based on distances, speed of travel, and relative location of various objects and places.

It occurred to me that what would be useful here would be a dynamic map. Such a map would allow placement of objects according to relative measurements, movement of objects according to specified speeds, and an animated display. In order to demonstrate Logo's usefulness in this humanities-related subject, I decided to implement the dynamic map.

The map is designed using an object-oriented style of programming. Each location object on the map has a position, a speed, a heading, and a shape associated with it. An object can be modified or addressed by first telling Logo that

you want to **TALKTO** a specific named object.

These object characteristics can be set with commands such as **NEWSPEED**, **NEWHEADING**, and **NEWSHAPE**. A speed is specified in distance (expressible in miles) per interval where an interval passes each time the **NEXT** command is executed (the default speed is zero). The heading indicates where the object is aimed. The shape is specified as one or more Logo commands used to draw the shape for this object on the screen.

The location can be changed with the **NEWLOCN** command. This command permits some readable, relative positionings to be specified. For example, **NEWLOCN MILES 10 NORTH OF LOCNOF "DETROIT** changes the location of an object to 10 miles north of the location of another object named Detroit.

The distances and locations of objects can be queried at any time with standard Logo **PRINT** commands.

As with the **CHEM** microworld, this dynamic map does not serve anyone's real purpose right now, but it demonstrates a program that might be useful to a person dealing with a real world problem in a humanities-related field.

Why Logo?

Probably the most pressing question in discussing these examples and others of Logo being used in the world outside of education is "Why Logo?" Why not some other of the hundreds, if not thousands, of computer languages in existence?

Let me outline some of the characteristics of these problems which made them prime candidates for Logo solutions.

- A microcomputer implementation was desired or required for most of these problems. Use of a microcomputer does limit the number of programming languages accessible.
- The problems required *ad hoc* solutions. The type of programs that were developed in these examples are not readily available in the software marketplace. If a user needed any of these programs, they would most probably have to write their own.
- None of the problems specified a need for fast execution time. Three of the four were designed for interactive use, where speed is limited to user's response time.
- A language particular to a specific problem domain was necessary in all of these contexts, and appropriate syntax (e.g., **PRINT DISTANCE BETWEEN LOCNOF "OBJECT1 LOCNOF "OBJECT2**, a valid command in the dynamic map) was a bonus. Three of the four examples were written for infrequent computer users in mind. These users could not be expected to learn inconsequential details like subroutines. They could be expected to learn a command set of reasonable, problem-specific words combined in reasonable ways - especially important since most users would not be using these programs most of their day so anything less would quickly be forgotten.
- For most of these examples, having a standardised graphics and music interface, built in to the language, was useful or necessary.

Logo filled these needs. Since Logo allows definition of new words, the language of the solution could be contoured to the problem. Logo is not fast in execution

time, but it is good for interactive applications. It provides an excellent development environment so it is inexpensive to quickly develop a solution to play and explore with. The turtle graphics and TOOT music primitives are standard and appear in almost every Logo implementation.

There are other languages that could fill these needs. Forth shares many of these characteristics, like problem-specific language, if the users are willing to use Reverse Polish Notation. APL would also be useful if the users are comfortable with the special symbols. Perhaps Lisp or Borland's Turbo Prolog would make good candidates.

The choice of Logo among these candidates was a personal one. I find it accessible, understandable, and usable. For problems I encounter that have requirements like those I describe above, I often use Logo in the solution.

References

- Becker, R.A. & J.M. Chambers. (1985). *S: An interactive environment for data analysis and graphics*. Belmont, CA: Wadsworth.
- Bentley, J. (1986, August). Programming pearls: Little languages. *Communications of the ACM*, 29(8), pp. 711-721.
- Harvey, B. (1985). *Computer science Logo style*. Cambridge, MA: MIT Press.
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books, Inc.

Video Job Center

Ray Hoag
Academic Computing Center
Grand Rapids Junior College
143 Bostwick Avenue N.E.
Grand Rapids, MI 49503
(616) 456-4430

Primary Associates:

Joel Burke, AMWAY Corporation
Sue Merizon, Grand Rapids Junior College
Jeff VandenBosch, Data Facts, Inc.
Joe Walsh, Visual Image Productions

Abstract

Grand Rapids Junior College (JC) has created the Video Job Center consisting of a personal computing system with videodisc, touch screen display, magnetic disk storage and automatic-teller-machine-like printer peripherals. The system is integrated into a kiosk which allows public interaction only through the touch screen display. The user receives printouts through a provided slot in the kiosk. Using the touch screen capabilities, users are able to view a video motion sequence, salary ranges, job growth projections and current Greater Grand Rapids job openings for any of eight major job areas. The Video Job Center was accomplished through joint efforts of community resource people, JC staff and JC Computer Applications Division practicum students. Is this a worthwhile activity for JC? Is this a worthwhile application of technology? If answers to these questions are positive, then are there ways to improve the Video Job Center and how is the transition made to other related worthwhile activities?

Description

What is the Video Job Center? This question is answered by describing the Video Job Center's physical characteristics, information capabilities and operation. Physically, the heart of the Video Job Center is an InteractV-2 interactive video and computing system unit by National Cash Register (NCR). It has 512,000 positions of internal memory and the ability to access 20,000,000 plus characters of information from magnetic disk storage. An electronic

interface card in the system unit allows for the accessing of video information (still frames and motion sequences) from a Pioneer LD-V1000 laser video disc player. Two audio tracks on the video disc player are computer controlled.

Another electronic interface card connects to an automatic teller machine (ATM) type of printer. Information is printed at 30 characters per second and 40 characters per line. A further electronic interface card allows for the connection of a 640 by 400 resolution color monitor to display text, graphics and video information. The video display monitor is also a touch sensitive input device. When a person touches the screen, the touch point is relayed to the computer's processing unit for programmed response. All of the physical components of the system are housed in a kiosk, a small structure that allows access only by using the video display's touch screen capabilities. A small slot in the kiosk provides printed information. All other components of the system are accessed only by unlocking panels on the kiosk.

Video disc information consists of still frames and motion sequences ranging in length from 30 seconds to two minutes. Included among the motion sequences are three different welcoming statements, a thank you for using the Video Job Center, general information on choosing a career, various instructional messages and these eight job areas:

1. Computer/Technology.
2. Health care.
3. Food services.
4. Office/business.
5. Sales.
6. Maintenance/service.
7. Manufacturing.
8. General occupations.

Over \$15,000 of the projects's \$26,000 budget were utilized for producing the video information and getting the video disc mastered. It is estimated that this figure would have been at least three times greater were it not for significant amounts of volunteer time and the sharing and inclusion of video that had been produced for other uses. Considerable design and planning went into this endeavor. The video information cannot be changed without editing a master video tape and going through a two thousand dollar laser disc mastering process.

One of the initial project activities was the preparing of a script. Several JC content specialists participated in this activity. This script is stored as audio information on the video disc. The script is on one audio track and background music is on the second audio track. In some instances a male voice is on one audio track and a female voice is on the second audio track. Information stored on magnetic media includes salary ranges and growth projections. Each job area is subdivided into job clusters. Production, preparation and service are example food service clusters. The salary and growth information is provided graphically.

Jobs for which a Greater Grand Rapids area employer currently needs an employee are provided by job area (not by cluster). These are actual jobs, not just examples. Because of the volatile nature of the salary ranges, growth projections and especially the jobs, this information is stored on changeable magnetic media. Initially job information will be updated once or twice a week. Future plans call for the use of electronic communications to update job openings as they occur.

Also stored on the magnetic media are the program (written in BASIC); the software drivers for the touch screen, video disc player and ATM printer; and several computer generated graphics. The graphics are stored on the faster access hard disk system. The smaller capacity floppy disk serves primarily to get the job information into the system.

An operational scenario might go like this. A person is attracted to the kiosk, not only by its signage and placement, but also by the fact that it "wakes up" every 30 seconds to play an audio and motion video sequence of welcome and encouragement. "Just touch me, I'll wait till you begin." A video table of contents is on the screen and the person touches his or her choice.

The audio responds with a positive phrase such as "excellent choice." A video motion sequence is played describing different jobs and their characteristics. After approximately one minute, the person chooses one of the job clusters in their chosen area. Upon touching the job cluster, a computer generated bar chart shows salary ranges. Touching the "continue" spot, the person is presented a second bar chart on growth projections. Touching "continue" again, the

person is invited to touch the screen to see a list of jobs in the job area.

Upon doing this, blocks are drawn on the screen each with a job title on it. The person touches a job block of interest and the system responds with a brief job description (comparable to an ad in the newspaper). The person is then given the choice of printing the job information, ending the session or going back to choose another job block. Upon touching the "print" selection, the ATM printer prints the job information and cuts the paper. The included instructions encourage the recipient to take this to the JC Job Placement Office to pursue the job. The person walks away from the Video Job Center. After 30 seconds of inactivity, the Video Job Center returns to the welcome message of its attract mode.

The team approach was used to accomplish this project. Project management, system design, video production, computer programming, management consulting, interactive video consulting and script writing functions were team member tasks. Team members represented a video production house, JC practicum students, community computer professionals as consultants and JC faculty.

Computer to Computer Communications

Richard G. Lewandowski
Science and Technology Specialist
Ypsilanti Public Schools
210 West Cross Street
Ypsilanti, Michigan 48197-1817
Voice (313) 482-6404 E-Mail (313) 482-8090

Many computer application users have advanced with technology to another machine or find themselves wanting to transfer information organized last year to this year's dream machine. Computer to computer connections can transfer this data. Learning how to accomplish this task can take a user to a new level of understanding of computer operations.

This article will discuss some of the techniques available to use in transferring data between various models and brands of computers. The type of information to be transferred should be ASCII, DIF, Quick File, VisiCalc, etc. based material. This material might be the type generated by word processors, database programs, or spreadsheets. It is assumed the reader will have a basic understanding of DOS operations on their own machines and have available the manuals or similar instructions for operation of their telecommunications software and hardware.

Generally, stand-alone processing on a microcomputer involves communications along several types of channels or lines. Disk drives, printers, a mouse or other input and output devices are connected by cables which provide for data exchange. Although the type connection these peripherals use varies, they all depend upon use of the proper cables and ports for signals to travel through.

The most commonly available type of connection or port is the RS-232. Standardized, it is available on every microcomputer in some form. Apple II's use a serial card in a slot to put the signals in the proper place. IBM type machines and others come with at least one serial port available. The Macintosh line depends on proper cabling to bring the signals out to standard connectors.

Since the signals coming out of these ports are standard, machines can use the same peripherals such as modems and printers which connect to these elongated "D" shaped connectors. While running a special category of software called communications software, computers can be made to exchange data with one another. Programs transferred by these methods will run only if they use the most generic of commands because of the subtle hardware and language differences that exist between all but identical machines. Text type materials can be easily transferred and utilized on nearly every machine.

One tool makes this job easy and solves a number of problems. A null modem cable needs to be placed between the two machines. This cable is specially constructed to switch the signals in pins 2 and 3 to allow the machines to listen and talk with one another at the right times. The Apple Super Serial Card has a block which may be placed with its arrow pointing

to modem or terminal. Switching the position of this block toggles the connection of those pins. Null modem cables are available everywhere and may be a simple block with two of the "D" connectors to connect the cables.

The communications software handles establishment of other protocols between the two machines such as speed or baud rate, parity, stop bits and other variables which must match. Speeds up to 9600 baud (the same as Apple II disk access speed) can easily be obtained.

An easier and less intimidating method of accomplishing file transfer can take place using modem to modem transfer of files. This technique is commonly used when an individual contacts a bulletin board or data information service such as CompuServe. Telephone connections need not be used, however. Machines in the same room can be united through a cord with modular jacks at each end. If the modems connected to each machine are connected in this manner and the modems are properly set, file transfer can take place. One machine must be set to originate a call, while the other answers. The answering machine, if equipped with a modem using the Hayes Command Set, can be set from within the terminal mode by typing "ATA" followed by a carriage return. The calling machine then initiates the exchange by being commanded to call.

Data files not available directly to the user, such as those from the old PFS series, can be transferred in the desired format with the appropriate information. Instead of running the cable to the printer, it may be directed to another computer's RS-232. That computer must be running a communication program to receive the data. A problem which may occur, depending on the size of the buffer storage area and the size of the file, is loss of data. When the receiving machine must write received data to disk, it may send an unreceivable signal to the transmitting machine commanding it to stop. Some software/machine combinations are able to buffer incoming data while handling disk operations. Another solution to accomplish this transfer and others, is to obtain one of the many public domain utilities that have been written to deal with the peculiarities of file structures in programs.

Within almost every advanced program there are utilities providing for the importing of files from other machines and programs. Appleworks, for example, will accept DIF, ASCII, Quick Files, and VisiCalc imports. Although these software features were designed to help with transfer of files made on the same machine, they will work with data transferred to the machine from another.

The last consideration is one of connectors. You may need to purchase a new cable or connector such as a null modem. Borrowing from another machine may work; a Macintosh Plus needs an Imagewriter cable to connect to a modem or another computer. Building your own requires time, careful use of tools, and pin connector information. Your best source of help, if you have specific questions to ask, may be your dealer, a local user group, or a bulletin board or conferencing system. Most people are willing to share their discoveries and some may have suggestions to the solution of problems they have not investigated.

The following page contains illustrations and examples of specific cables and pin connector information.

SPECIFIC CABLE AND PIN CONNECTOR INFORMATION

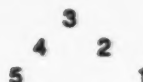
A NULL MODEM

in — out
2 — 3
3 — 2
7 — 7
20 — 6+8
6+8 — 20
4+5 — 4+5

APPLE II-C DIN TYPE CONNECTOR

DIN 1— Data Terminal Ready
DIN 2— Transmit Data
DIN 3— Ground
DIN 4— Read Data
DIN 5— Data Set Ready

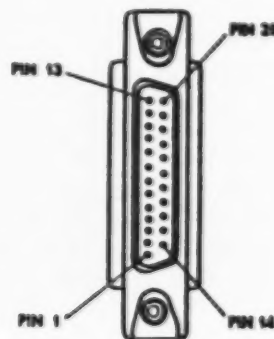
APPLE II-C DIN CONNECTOR CONFIGURATION



MODEM END OF RS-232 (Hayes Standard)

1 — protective ground
2 — transmit data
3 — read data
5 — clear to send
6 — data set ready
7 — signal ground
8 — carrier detect
20 — data terminal ready
22 — ring detect

RS-232 / DB-25 CONNECTOR DIAGRAM



AT 9-PIN CONNECTOR

1 - signal ground
2 - receive data
3 - transmit data
4 - carrier detect
5 - ground
6 - data set ready
7 - data terminal ready

Computer and the Arts

Scott Michael Lis
Computer Teacher

Fall
Croswell Lexington
High School
Croswell, MI 48422
(313) 679-3555

Summer
Interlochen Center
for the Arts
Interlochen, MI 49643
(616) 276-9221

Computer programming is being taught everywhere, but how often are the graphics and sound features used in the classroom? This paper will describe activities for each, and the exercises can be extended and expanded to provide enrichment to the computer programming class by using these major marketing features of the microcomputer.

There are plenty of graphics and music programs available for each of the different brands of computers. My goal is to teach programming with a minimum of expense and a maximum challenge to the student. The graphics and sound commands available on all microcomputers can be used without extra cost and they can be used to teach difficult concepts in a creative, rewarding and challenging manner.

Shape Table Animation

Shape table animation requires the knowledge or learning of the high resolution graphics commands, INKEY\$ or GET statement, the IF-THEN construct, and the GOTO statement in BASIC, or similar commands in Pascal. The DRAW statement may also be available but is not required. When using high resolution graphics on the Apple II computers some points may not show up. This is because of the way colors are generated. A rule of thumb is to use even points with even colors and odd coordinates with odd colors.

Statement of the Problem

Using high resolution graphics:

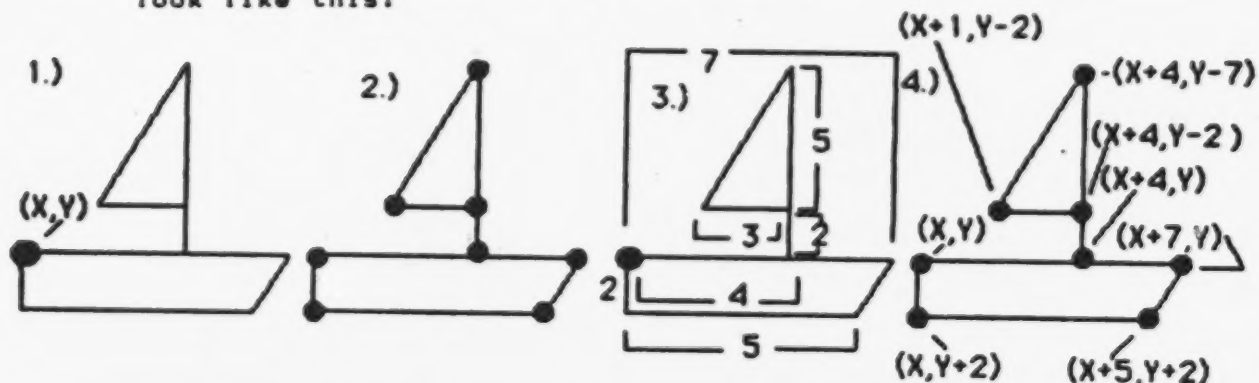
- 1.) Draw a picture on the screen (sky, road, room)
- 2.) Construct an object or shape relative to any starting point (x,y) on the screen. (ship, car, bird)
- 3.) Animate the object by drawing and erasing it with a different (x,y) starting point. Control the direction with the keyboard.

The picture must be mapped out on graph paper before entering it into the computer. The object can be defined using the Apple shape table routine and some comments on this follow the next section.

The shape can be drawn using high resolution graphics, but it must not use any absolute points. Each point is relative to an (x,y) starting point.

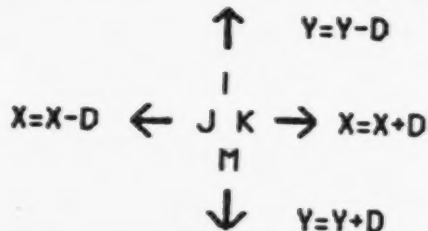
- 1.) Label the starting point (x,y) on a sheet of graph paper and construct a simple line drawing of the object.
- 2.) Place dots at all of the endpoints
- 3.) Write in the length of each line segment
- 4.) Write the coordinates for each endpoint relative to the (x,y) starting point. Use the lengths from step 3 to figure the horizontal and vertical distances.
- 5.) Write the code

If $(0,0)$ is in the top left corner, the steps may look like this:



If you are using the method described above or you do not have an Apple II skip this paragraph. If you are using the Apple II you may want to use the shape table routine available. This method is complex, but it has worked with 7th-12th grades and the students will benefit by using binary and the machine language necessary for this routine. Also the ROT and SCALE commands can be used. To use shape tables consult one of the many programming manuals. Remember to start simple, use a single box-like shape with no curves or angles other than 90 degrees. Avoid using nonmoving vectors because the zeros can end the definition prematurely. Write the 3 bit vectors in a column, mark the groups that will form each byte, and write the bytes with the top most vectors to the very right.

Once the picture and the object are drawn on the screen the keyboard routine should be written. The following diagram shows how to write equations to translate or move the object around the screen. The diagonal directions can be obtained by varying both x and y.



The directions I, J, M, K are entered via the INKEY\$ or GET command for single key entry with no echo to the screen. Any other character should end the program. The new coordinates must be tested with the screen boundaries and adjusted if they are out of range. If you are using Turbo Pascal the getdotcolor procedure can add some interesting effects. The object is erased by drawing it in the background color. The nine steps are as follows:

- 1.) Draw background
- 2.) Draw object
- 3.) Wait for a key to be pressed
- 4.) Erase the object
- 5.) Check for a valid key
- 6.) Calculate the new coordinates or end

- 7.) Check and adjust if the object is out of bounds
- 8.) Draw the object in the new location
- 9.) Goto step 3

Introduction to Files Using Musio

This activity requires the knowledge or learning of how to read, write and check errors with the disk, the INPUT, INKEY\$ or GET, IF-THEN, GOTO or GOSUB, FOR-NEXT and music commands in BASIC or similar commands in Pascal.

Statement of the Problem

Using a menu or command line, request and complete the following activities:

- 1.) List song names
- 2.) Play a song
- 3.) Write a song to the disk
- 4.) Enter or change a song
- 5.) Read a song from the disk

The student does not need to be able to read music, the exercise works just as well if the students use songs already translated like the following example:

FREQUENCY:	222	214	206	214	222	222	222
NAME:	A	G	F	G	A	A	A



DURATION:	10	10	10	10	10	10	20
-----------	----	----	----	----	----	----	----

To further simplify the project, the notes can be entered directly into the program with data statements and written to the disk files later.

Note Entry

- 1.) Input the pitch, zero to end, -1 for a rest
- 2.) Add to the array count

- 3.) Input the duration, zero to end
- 4.) Add to the array count
- 5.) Check the range and request change if needed
- 6.) Play note
- 7.) Goto step one
- 8.) If the end is selected the user should be allowed to change any note by entering the array index number.
- 9.) The number of notes in memory should be saved.
- 10.) Return to the menu

Playback

- 1.) Use the note count as the upper bound in a loop and "play" the pitch, duration arrays back. If you are using the Apple II, you will need to enter a speaker driver routine. This can be found in several programming manuals.
- 2.) Return to the menu

Write to the Disk

- 1.) Input the song name
- 2.) Open the new file, or first delete the old
- 3.) Write the note count as the first line
- 4.) Write the pitch on the next line
- 5.) Write the duration on the next line
- 6.) Repeat steps 4 and 5 for all the notes
- 7.) Close the file
- 8.) Return to the menu

Read from the Disk

- 1.) Input the song name
- 2.) Open the old file, or reset if not found
- 3.) Read the note count line
- 4.) Read a pitch into the array
- 5.) Read a duration into the array
- 6.) Repeat 4 and 5 for each note
- 7.) Close the file
- 8.) Return to the menu

An Educator's View of Software Piracy

**Sharon Octernaud
Microcomputer Software Trainer
Office of Academic Computing
Ferris State College
204 Prakken
Big Rapids, MI 49307
(616) 796-0461 ext. 4120**

Stealing is running rampant in our schools, colleges, and universities under the guise of software piracy. The illegal copying of software poses large, difficult, emotional, and philosophical questions. Some observers even think this common practice of stealing software is endangering our ethics. (Lewenstein, p. 181) There must be an increase in societal awareness of the mutual obligations of computer software users and developers. "In almost no country is there a clear law protecting new programs from being copied." (Martin & Norman, p. 466) Currently the development of hardware has far exceeded the development of software and to encourage further software development, some form of legal structure providing an incentive to inventors and entrepreneurs is needed. As well as achieving some financial gain for the fruits of their labors, software authors have the right to preserve the identity and dignity of their work. (Greenberger, p. 171) The laws, however, must not only protect the rights of software developers but also allow the public to have access to new information and knowledge. (Johnson & Snapper, p. 322) Writing laws to benefit both the developers and the public is a complex process.

What a computer program is determines how the law can and will protect it. Therefore, before viewing the kinds of legal protection accorded software, a basic understanding of what constitutes a computer program is desirable. A computer program is a set of instructions which tells the computer how to do its work. Several steps are involved in writing a program. First the issues to be handled must be identified and formulated. Next a solution in the form of an algorithm must be clearly expressed, usually with the aid of a flow chart. The flow chart is used as a guide for putting the algorithm in computer language which is known as code. Once the program has been written in code, it becomes a source program and can be translated by a compiler into machine language. When expressed in machine language, the program is called an object program. It is actually the object program which enables the computer to solve a given problem. With so many complex steps involved in writing a computer program, it becomes evident why laws protecting such programs are ambiguous. "What is the program? If the process were stopped at any one stage, would the program as it existed at that point be entitled to legal protection?" (Johnson & Snapper, p. 307)

Because discussion generated by these two questions involves not only the fields of science and law but also philosophy, the issue remains unresolved. A myriad of patents, copyrights, trade-

secrets, licenses, and leases exist. Attorneys specializing in the computer field are unable to offer clear-cut answers. Well-defined laws would not only protect the software developers but also the software users. Software piracy actually is fostered by legal ambiguity as individuals are unsure of their rights as software users. The future will add to the complexity of the legal issue. More programs are likely to be converted to hardware and stored as a part of the computer itself. With the increased power of machines, a programming language will resemble normal speech. If the nature of a program can no longer be analyzed, how can it be patented? (Johnson & Snapper p. 325)

For most software users in an educational setting, the single user/single machine concept is a safe one to follow. This concept permits a single backup copy of the program to be made for security purposes but implies that only one person may use the program at a given time. This prevents one copy of a software package from being used on several computers by multiple users. As a book can be read by only a single reader at a time, so also a computer program can be used by only a single person at a time. There may be instances where institutions enter into special licensing agreements with a company thereby permitting multiple users of a given program. Those directly in charge of computer labs will be aware of any arrangements of this nature. In an effort to cut down on software piracy, many companies have included copy-protection schemes in their software making it difficult to copy programs. Such copy-protection measures tend to alienate customers. In fact, Craig Stark, editor of PC Magazine, will not purchase a copy-protected program unless he knows that he can make as many backup copies of it as he deems necessary for his security. (Stark, p. 82)

It is impossible to determine how big the software piracy problem actually is. Experts estimate that one to three pirated copies exist for every legal copy of a program. Jerry Dreyer is the president of the Association of Data Processing Service Organizations (ADAP-SO). This organization is currently leading an industry campaign against software piracy. Dreyer estimates that 50 percent of the business software market is lost to pirates. (Lewenstein, p. 181 & 182) The research firm Future Computing supports Jerry Dreyer's estimate that 50 percent of all business software is unauthorized. In fact, this research firm estimates that piracy cost software publishers \$1.3 billion in lost revenues between 1981 and 1984. Future Computing projects losses for 1985 to total \$800 million. (Bunnell, p. 18)

Dreyer believes that corporate policies never actually condone piracy. However, he feels that unless industries issue strong corporate statements against piracy, many managers may not think about the effects of their actions. By Dreyer's standards, this lack of reflection implies that the business world is neutral in its manner of dealing with software piracy rather than being unethical. However, others in the computer field fear that this neutral stance conditions leaders of the next generation into the acceptance of thievery. (Lewenstein, p. 184)

Howard Stevenson, a professor at the Harvard Business School, believes that justice may be on the side of the pirate. "Why am I being charged \$600," he asks, "for something which costs \$13?" (Lewenstein, p.186) Stevenson is one of many software users who protest against an industry that is trying to make a quick profit.

Fritz H. Grupe of the University of Nevada - Reno strongly believes "that all software utilized on University-owned equipment and within University-owned facilities be fully and correctly licensed." (Grupe, p. 9 & 10) His university is committed to ensuring that all administration, faculty, staff and students employing microcomputers on campus use software within the confines of the law. Because of increased lawsuits against organizations using pirated software, UNR views that failure to adhere to such a policy jeopardizes the university's financial position. In addition to UNR's financial concerns, lies a commitment to the moral and ethical bases for a high quality educational process and research environment.

Thomas Dunfee, a professor of social responsibility at Wharton Business School, believes that "piracy destroys the economic system. It makes the system less efficient. Honesty is more efficient." (Lewenstein, p. 186)

If ethical decisions are not based on great moral ideas but on profits and gains and in strengthening the American economy, a solution must be found which offers hope to those who believe in honesty and side with the software publishers' right to make money. (Lewenstein, p. 186) Some software developers in an attempt to find a solution to this philosophical problem are marketing a new form of software which is distributed as freeware/shareware. I believe this marketing technique may well be the best approach available to discouraging software piracy.

In order to better understand this concept, an awareness of three software categories is essential. The first category of software is public domain. This software is free to everyone. The authors of these programs have relinquished their rights to control the distribution of their programs. These authors have written programs which have been of use to them and they are happy to share their work with others. Public domain software is widely circulated by computer clubs and electronic bulletin boards. The second category of software is freeware/shareware. This software is free to acquire and share with others. However, users are requested to send the author a monetary contribution if they find the program worthwhile and use it. With this type of software, the moral obligation rests with the user rather than with the distributor. User support groups and electronic bulletin boards distribute the software free of charge. The third category of software is proprietary. This software is sold and licensed commercially and is legally protected in a variety of ways. Proprietary software is distributed by manufacturers to retailers who in turn pass it on to computer stores and mail order catalogs. Needless to say, each of these businesses is advertising and selling products in order to

make a profit. The purchase price of the software increases within each step of the distribution process. The freeware/shareware concept works both for the users and for the authors. There are no copy-protection schemes making it difficult to backup copies. Users are able to try a software package to see if it meets their needs before purchasing it. Freeware/shareware software generally costs less than \$100 per program. Equivalent proprietary software packages are sold for \$300-\$400. Many honest users willingly register their copies to encourage this form of marketing. The software developers are not paying commercial retailers a percentage of the profits and commercial advertising is kept to a minimum. It is estimated that an average developer earns between \$40,000 and \$50,000 a year per program by marketing software in this fashion. ("Q & A Corner", p. 9)

Another deterrent to software piracy is centralized purchasing and inventory control of software. By adopting creative measures to entice customers into buying a program, the computer industry can combat piracy. If retailers offer licensing agreements facilitating volume discounts and offer more after-sale support and service, piracy will no longer be as alluring. The solution to software piracy lies in educating the public. As the computer is new to daily living, people do not comprehend that it is wrong to pirate software. (Taylor, p. 167) Mitch Kapor is chairman of Lotus Development Corporation, a major microcomputer software vendor. He uses shoplifting as an analogy to piracy. Both of these problems can be addressed by a mix of technical and legal methods. However, he states that "only a strong climate of public opposition to piracy as a moral issue will be sufficient to bring the problem under control." (Kapor, p. 239) Although the current laws are unclear, I feel that educators have a moral obligation to refrain from pirating software and to discourage others from doing so in the classroom and at the work place. Students need to be educated as to the ethical and economic implications of such a practice. By finding good usable software in the freeware/shareware/public domain categories; and by widely disseminating it, offering training for it, and adopting it across campuses, piracy can be lessened. If universities chose to reward their instructors for software development as they do for publishing, this would serve as an additional incentive for instructors to write computer programs. It is very likely that these programs would be distributed as freeware/shareware/public domain software. It is my belief that as more and more good software becomes available in the freeware/shareware/public domain categories, the general public will become more aware of its existence. Instead of purchasing a proprietary piece of software, other alternatives will be explored before selecting a commercial package. Freeware/shareware software will be used increasingly in the business world thus changing current accepted norms. Rather than developing increasingly complex copy-protection solutions, freeware/shareware/public domain software offers a viable solution to the software piracy issue.

References

- Adams, J. Mack, and Haden, Douglas H. *Social Effects of Computer Use and Misuse*. New York: John Wiley & Sons, 1976.
- Breslow, Jordan J. "Caveats for Custom Software." *PC World*, March 1985, pp.227-235.
- Bunnell, David. "Software Piracy--One More Time." *PC World*, May 1985, pp. 11-18.
- Greenberger, Martin, ed. *Computers, Communications, and the Public Interest*. Baltimore, Maryland: The John Hopkins Press, 1971.
- Grupe, Fritz H. "Caught Red-Handed: Severe Penalties for Software Piracy." *SIG Bulletin*, Jan./Feb./Mr., 1986, pp. 17 & 18.
- Johnson, Deborah G., and Snapper, John W. *Ethical Issues in the Use of Computers*. Belmont, California: Wadsworth Publishing Company, 1985.
- Kapor, Mitchell. "Piracy and Software Protection." *Creative Computing*, November 1984, pp. 236 & 239.
- Kennedy, Don. "Software Ripoff." *PC Magazine*, March 19, 1985, p.34.
- Lewenstein, Bruce V. "The Ethics of Software Piracy." *PC Magazine*, April 30, 1985, pp. 181-187.
- Martin, James, and Norman, Adrian R.D. *The Computerized Society*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1980.
- Mayo, Ken. "Business Battles Its In-House Pirates." *Business Computer Systems*, February 1985, pp. 60-65 65.
- "Q & A Corner." *PC-Sig News*. September 1986, p. 9.
- Sanders, Donald H. *Computers Today*. New York: McGraw-Hill Book Company, 1983.
- Stark, Craig. "Are They Clean?." *PC Magazine*, January 22, 1985, pp. 79 & 82.
- Sterne, Robert G., and Saidman, Perry J. "Copying Mass-Marketed Software." *Byte*, February 1985, pp. 387-390.
- Sterne, Robert G., and Saidman, Perry J. "The Sale of Computer Products." *Byte*, May 1985, pp. 399-412.
- Taylor, Jared. "The Copy-Protection Wars." *PC Magazine*, January 14, 1986. pp. 165-182.

**Michigan Occupational Data Analysis System -
Assessment Systems**

**Dr. Chris M. Olson, Associate Professor
Ms. Carol E. Culpepper, Curriculum Specialist**

**Michigan State University Occupational Research and
Assessment Resource Team
Department of Agricultural and Extension Education
102 Wills House
Michigan State University
East Lansing, Michigan 48824
(517) 353-0661 or 1-800-831-7523**

Data Base programs that reflect labor-education linkages are critical to the development of effective vocational education programs. While much has been written about data bases which facilitate specific program components, such as student aptitude and interest testing, very little has been written about those that promote access to information that is applicable to planning and development of vocational programs in their entirety. The reason for this is because it was not until 1983 that such a data base existed. This data base was developed in Michigan and is called the Michigan Occupational Data Analysis System - Assessment Programs. It is comprised of two data bases, the first of which permits persons to access information to help them in the areas of guidance, curriculum and decision making. The second data base, assessment, will help educators with assessing the progress that students make within the various occupational programs.

Let us explore this further. The Michigan Occupational Data Analysis System contains three major data bases. These data bases are interrelated in the program. The first data base is the abridgement of curriculum developed by the Michigan Department of Education and the Vocational-Technical Education Consortium of States. It contains a comprehensive listing of occupational duties and tasks. These lists provide educators with information that aids design of occupational programs which represent needs identified by employers.

The program's second data base is a derivative of the Handbook of Occupational Keywords. The Handbook, which was developed by the Department of Labor Jobs Services,

contains descriptions of aspects of work that are germane to specific occupations as well as descriptors that are relevant to several occupations. In MODAS, the descriptors are used to expedite searches across occupational boundaries. As a result a search is performed across occupational areas, educators can identify duties-tasks that are common to more than one program area.

The program's third data base reflects information from the Vocational Preparations Occupations Handbook, National Occupational Information Coordinating Committee. This data base cross indexes job titles with Classification of Instructional Program Codes. In addition, estimated length of training time, physical-environmental factors and computational and language development levels are referenced to each job title that is listed for a program area. Further, codes (SOC) that classify occupations into homogenous groups and codes (OES) which reflect employment supply-demand factors are a part of this data base.

Information from the third data base assists educators with making effective decisions about the inclusion or exclusion of occupational programs. For example, decision makers can respond to questions such as: Are there employment opportunities in this occupational area? What careers are feasible to offer in a program area? Is the scope of the program sufficient to facilitate the needs of the students with varied learning capabilities? Is there a need to implement additional instructional support systems that will aid students in experiencing success in a program? Do the program components represent a homogenous grouping of work activities?

MODAS also contains data from a fourth source, the Handbook for Analyzing Jobs. This document provides the MODAS program with data that reflects various occupational characteristics. For example, aptitudes and related General Aptitude Test Battery scores that are associated with a given job title as well as codes workfields and Materials, Products, Subject Matter, Services (MPSMS) which categorize work performed on the basis of how work gets done and what gets done as a result of the work activities have been matched with the other labor-education references that are covered in the MODAS program.

Educators can use these data to evaluate the extent to which learning activities promote aptitudes associated with an occupational area, to assess the degree to which the program facilitates continuous learning and to identify program outcomes.

In summary, the MODAS' merger of external factors with educational program development factors enables educators to design, adopt, implement and maintain programs that meet the needs of students and business and industry.

The second Occupational Data Program, assessment, was established in order for Michigan to comply with the section of the Carl D. Perkins Vocational Education Act of 1984 P.L. 98-524, that emphasizes vocational program improvement and evaluation. Educational agencies can use the computerized test item program to generate occupational competency assessment tests that are directly related to curricula taught in their vocational programs.

The program contains both written and performance test items. To date, the items have been received solely from states who are members of the Vocational-Technical Education Consortium of States (V-TECS). V-TECS is an organization of twenty-three state departments of Vocational-Technical Education who have united to share resources in developing curriculum and related products. Each member state has to contribute products each year as a condition of membership. Many of the V-TECS members posit, that standardized occupational competency tests are not appropriate for their programs. The primary reason for this thinking is that there is not a standard curriculum which is taught nationally. Therefore educators in these states believe curriculum is really a function for state and local levels where teachers and advisory committee members work together to identify those tasks which will make program participants employable. Consequently, V-TECS chose to support a test item bank containing items coded to individual duties and tasks such as those that are contained in the V-TECS task list. Because the items in the test item bank program are coded to individual duties and tasks in V-TECS catalogs, educators can select those duties and tasks that reflect the content of instructional units or modules, grading periods, semesters, and so forth, and develop tests to measure student competence within those units.

The computer program for the test item bank allows one to save each of these tests and later merge them into an exit or completer tests. Exit and completer testing are similar and different only in purpose. Exit testing is any testing performed as a student exits a program. This could be done at the end of the year or at any time during the

year when the student leaves the program but does not complete the program. Exit testing can be used to determine competence for grading purposes.

Conversely, completer testing is used when the student completes the program. Again, this could be at the end of the year or at any time during the year, as long as the student has completed the program of study. Completer testing can be used for grading purposes or for providing the student with an achievement record for personal records or job applications.

The program facilitates pre- and post- testing. Thus educators can assess student competence or proficiency before and again after instructions. This allows for documentation of progress, or a lack of progress made by students. Pre- and post- testing techniques can also be used in conjunction with unit, module, grading period or semester testing, exit testing, or completer testing.

The Michigan Occupational Test Item Bank Program contain both multiple choice, written and performance items. The multiple choice items are cataloged and filed by: cognitive level, C1=knowledge recall, C2=knowledge application, if the item related to safety and if it measures employability skills (ES). Performance items are developed around the V-TECS catalog page. These sections of the test identify the duty and task to be performed and the conditions under which the task will be performed. Further, a checklist of the steps required to perform the task is provided for the evaluator along with spaces to record the identity of the student taking the test, performance level of the student, additional criteria, product evaluation, and comments to explain special circumstances involving the student.

The Michigan Occupational Test Item Bank was field tested on students during the spring, 1986. Preliminary results from this field test indicate that tests developed from items randomly selected from the item bank are highly reliable because the item bank allows teacher to select items representative of course content, test validity is assumed.

In summary, the Michigan Occupational Test Item Program and supporting computer software installed on the mainframe computer at Michigan State University was designed to meet the varied needs of vocational educators. As a result, these educators can develop tests around their unique curriculum content. In addition, the educators can save

tests, modify tests, merge tests, and have tests scored using Michigan State University scoring service. This data base in conjunction with the MODAS data base can be used to develop customized vocational programs that meet the needs of students and the needs of the business and industry community.

Data Management to the Aid of Discovery

Mike Rains
Computer Literacy Instructor
Romulus Jr High
37300 Wick Rd.
Romulus, MI 48174
(313) 722-1050

Can a computer literacy course convince students that they can find a personal use for computers?

A year and a half ago the extent of my applications unit was to teach students how to build a database and to quiz them on their memorization of various commands. Students filled databases with names, addresses and phone numbers, brand name shoes, lost and found from the office, teacher ratings, boy and girl ratings, championship wrestling records...all pointless excuses for sticking information into slots. Once the data was entered there was nothing to do with it, there was no direction, no question to ask and answer, no purpose except to "go through the motions" of setting up a database. Worse yet, the information was probably more accessible and useful on paper! The computer's novelty hid the fact that students were performing a machine form of "busy work". I do not think they understood the purpose of data management software. Moreover, I do not believe they were convinced the computer had any value for them.

Since I wanted the computer to become a tool for students, I had to rethink my entire approach to learning in the classroom. Following Olds' (1986) lead, I now try to help my students argue more powerfully, to inform and persuade an audience about something he or she thinks data reveals. The computer becomes a "research assistant", indispensable to the young scientist or journalist.

Last spring, the success of our track statistics database pointed the way. Track team members and their friends used to crowd around the computer printout near the door in order to see the latest times, distances, point totals, and broken records. Soon the "next-day" printouts created a friendship between the team and our computer class. My students worked twice as hard to get reports out on time. Format errors were a crisis!

Out of this experience, I have developed a teaching strategy that replaces technical emphasis with a new focus on decision-making (McClelland, 1986). First, I FIND COMMUNICATION CHANNELS in the classroom, school, and community. Then I DEVELOP A SOCIAL ENVIRONMENT AROUND SPEAKER AND AUDIENCE sustained by a sense of obligation. And finally, I make sure the students INFORM THEIR AUDIENCE on a topic of interest. Developing 20 or more projects at the same time, students will see many ways of using data management software. Therefore, they will understand it better. Students show each other the power and usefulness of

the computer. They compare notes on how they can organize arguments with a computer (Olds, 1985). And most important of all, students use the computer to explain some part of the world to each other (Newman, 1986).

INTRODUCING APPLICATION PROGRAMS

I introduce data management and word processing programs by making them serve students' interests immediately. We create a database containing information about each student and exchange it with another school in order to introduce electronic pen pals. For fun we research the difference between the classes (music, TV, and movie tastes). Telecommunicating "informed" letters back and forth is our goal. The computer simply facilitates the effort. The rationale for introducing a course this way is that students learn only those managing skills necessary to accomplish something (Olds, 1986).

PRODUCING A MAGAZINE

To deepen students' understanding of data management programs we produce a research magazine. Individual projects move from the building of a database to the development of an original article. Interestingly enough, students assume the project is over when they are done entering data. The printing of a magazine pushes them further because it gives them not only an audience but also an opportunity to connect their work to a larger project. To emphasize the collective nature of our efforts we have purchased "press hats" with the magazine's name on the front.

An article is printed in our magazine only if, in the opinion of the teacher and other students, it informs the intended audience. Each article is brief and highly structured. The writer presents a portion of the database to support a claim, along with an explanation of how the data argues the case. Students also list further questions that could be researched (McClelland, 1986). And finally, the student acknowledges those who made the data available.

Projects work best that appeal to a student's desire to: learn about other people, discover things for themselves and inform others about them, help others in some way, contribute to management of the classroom, gain respect of their peers, and work together as a team.

Titles of some of our current projects are: The Most Dangerous Intersections of Romulus, Cemetary Stones and Changes in Lifespan, Speaker Pool for Computer Class, Natural Occurrence of Chemical Elements in Sea, Air, Earth Crust, and Moon, Trends in Death by Disease of the 20th Century, Inventory and Maintenance Records of the Lab Computers, Nuclear Power Plants Within Two States of Michigan, Girls Basketball Statistics, Romulus Crime Trends, Poverty and Corporate Investments in Central America, Personal Catalog of

New Words, The Best VCR Buy, Programming Team Scoreboard, Court Cases Involving Teenagers, Student TV Watching Habits, and Animal Metabolism and Heart Rate.

We use encyclopedias, almanacs, 1980 regional census data (Wayne State, 1985), library pamphlets, magazines, and other compilations of data. Lesko (1984) has cataloged government and commercial databases along with mailing addresses, phone numbers, and hints about how to use the Freedom of Information Act.

Our magazine gives me a conscious format for criticizing and praising students. Opportunities open up for introducing fallacies of argument like over-generalization and mistaking strong correlation for cause and effect. I hope producing a magazine stimulates learning of the investigative process better than research isolated in a classroom.

Some students present their articles not only to our class, but to science, social studies, and English classes as well, making a special effort to show how they used the computer to get results. Our school's wide-screen projector helps the audience see the computer's output. Teachers, coaches, administrators, and parents who see our magazine respect the efforts of the class and are more inclined to bring in data and request research. The tax-paying community welcomes the chance to find out what students are learning in computer class.

BUT KIDS HAVE TROUBLE COMING UP WITH A PROJECT

We solved this problem by developing a database of 100 project ideas, stored on the back of each student's data disk. I provide a brief description of the area the projects cover and then sort them into interest areas (i.e. science, student interest, language arts, community interest, class or school administration, social studies, etc.). Each project is also marked for its difficulty level and for its appeal to introspective or outgoing personalities. Next, I specify the audience in order to introduce the social setting. I enter the data source and provide an "inquiry question" to guide the students as they think about their problem and decide what information they will need to solve it. I also suggest a few categories to help students see the foundation of their database.

TEACHERS DO NOT HAVE ENOUGH TIME TO GET A PROJECT LIKE THIS OFF THE GROUND

We invite anybody to USE AND CONTRIBUTE TO OUR WAREHOUSE OF IDEAS. This would be a way to put a database to use...for teachers.

Another timesaver is to appoint a team of students who can answer "how do I do this?" questions about the software. This way the teacher is free to concentrate on the cognitive

questions involved in data exploration. When given their research question, students will not immediately see that they can answer it by alphabetically and numerically re-arranging data and searching records according to a criterion.

I have found a good source of ideas in the University of Michigan's CONFER tele-conference called CRLT:EdMICROS. In this conference look for a discussion item on classroom data management and the latest in magazine making.

LOCATING AND ENTERING DATA TAKES TOO MUCH TIME

Bring as many data sources into the classroom as possible. Photocopying pages ahead of time is best. Discourage students from choosing projects where collecting data is a chore. I have learned that the farther from the classroom students go to collect their data, the harder it is. Store data in a central location--books on a shelf, papers in file folders labelled by subject, and spare copies in a drawer. Mix in some finished databases where students can add their own records. AppleWorks and PFS data files can be converted from one to another by using a program called BRIDGE published by Artrax.

Data gathered by the school for other reasons can be a good source. This fall our school conducted a survey of 800 students' out-of-school interests in an effort to determine what clubs ought to be formed. Our class entered the data into a spreadsheet and tabulated the results. We reported our conclusions in the lead article of our first magazine.

Since my students like to do research on the student body, I include their questions on questionnaire administered by homeroom teachers once per semester. Data generated by such a survey can support many database projects.

Telecommunicating is an excellent way to streamline data collection. Students learn quickly how to download from online database services--DIALOG, COMPUSERVE, etc. However, high costs cut down the number of possible searches.

CHAOS INCREASES WITH THE NUMBER OF PROJECTS

Assigning students to research in groups and then relating group projects to one theme makes organizing easier and tends to unify the magazine. But, rather than the sheer number of projects, it is students' inexperience with building arguments that causes most of the confusion. Again, strategies have to be developed so that the teacher is totally available to answer questions about data relevance, research technique, and reasoning, because these questions are pivotal to a student's mental development. If the teacher is not in a continuous discussion with students, many projects will fall into ruin.

STUDENT EVALUATION

Watching students at work and observing how much interest they have in each other's discoveries provides a clear picture of how the unit is working. Generally, when things are going well students volunteer, a sign they feel liberated. In addition, I offer students an evaluative instrument at the end of the course, so that they can articulate their opinions on the main issues of the class. In this way the magazine becomes more student-directed over time (Newman, 1986).

CONCLUSION

At a time when grades and job training are the predominant means of motivating students, attempts at shifting the weight to the present are bound to be awkward at first. But every small gain is worth it. When students leave my class, I hope they look back and feel they have used the computer to discover new social ties and intellectual realms.

REFERENCES

- Lesko, M. (1984). THE COMPUTER DATA AND DATABASE SOURCE BOOK. New York: Avon Books.
- McClelland, J. (1986, Spring). A New Twist on an Old Skill: Retrieving Information With Computers to Enhance Decision-Making Processes. COMPUTERS IN THE SCHOOLS, 3(1), 83-88.
- Newman, J. M. (1986, March). Online: Using a Database in the Classroom. LANGUAGE ARTS, 63(3), 315-319.
- Olds, H. F. (1985, Spring). The Microcomputer and the Hidden Curriculum. COMPUTERS IN THE SCHOOLS, 2(1), 3-14.
- Olds, H. F. (1986, Spring). Information Management: A New Tool for a New Curriculum. COMPUTERS IN THE SCHOOLS, 3(1), 5-22.
- Wayne State University School of Business. (1985). MICHIGAN STATISTICAL ABSTRACT. Wayne State University.

Bibbits, Gribbits and other Logical Things for Grades 1-3

Dr. Charlotte Scherer
Director, Clinical Experience and Computer Labs
College of Education and Allied Professions
Bowling Green State University
Bowling Green, OH 43403-0243
(419)372-7392

Introduction

For at least the last two years now, problem-solving has been a key word in educational computing. Computing magazines have had numerous articles on this subject and catalogues of educational software have devoted entire sections to "problem-solving" software. In fact, one author has adapted activities from a computer programming reference manual to provide students practice in problem solving strategies in math (Irby, 1985). Another article summarizes a study which showed that both sensing and intuitive students, as measured by a Myers-Briggs Type Indicator test, preferred problem-solving CAI programs to drill and practice ones (Parker, Barry & Exner, 1984). Believing that using problem-solving software might be a better use of the computer's capabilities and that perhaps teaching such skills could be done more easily with a computer than with the more traditional educational resources, this author tried two of the more highly touted pieces of software that claim to teach problem-solving skills to young children. An enrichment course was offered through Bowling Green State University's Continuing Education program to 1st, 2nd, and 3rd graders and taught by this author with the aid of trained volunteer teacher education majors.

The Project

The course was taught three times, once with six sessions of one hour per week, once during the summer with five sessions of one hour for each of five consecutive days, and once with seven sessions of one hour per week. All were taught in a small campus lab with 9 computers and one large monitor available for demonstration purposes. The software programs chosen were Moptown Parade by The Learning Company/Addison-Wesley and Teddy's Playground by Sunburst (Grimm, 1981; Stanger, 1985). For fun on the last day of two of the sessions, a little bit of Logo turtle graphics was introduced. The groups of children who took the course were small: 7, 10 and 10 which meant time for lots of individualization, especially with a good adult-child ratio two of the times the course was offered. The first offering the ratio was 6 adults and 7 children; during the summer, there was only one university student, a philosophy graduate student for a ratio of 2 to 10; the third time there were 8 university students and 10 children, which meant that some assisted two children, and the author who directed the entire program was free to circulate and assist as needed or to observe. Goals for the programs were several: 1) to observe the children's reactions to the programs; 2) to determine whether the children would become adept at the skills of defining and identifying traits, using problem-solving, making twins and opposites, finding differences, identifying patterns, and using deduction to decide members of a group; 3) to determine whether the children could transfer the skills learned to reading and math; 4) to determine whether the children enjoyed working with computers to learn educational skills through a game approach; 5) to determine whether these programs, which come with extensive teacher materials, would be easy for a teacher to learn and use with children; and 6) to provide a learning experience for the volunteer teacher education majors in working with young children and computers. For the first offering, the children were tested at the end using paper and pencil exercises similar to those on the supplementary worksheets and a questionnaire to evaluate the accomplishment of the goals. For the second and third offerings both a pre- and post-test were given, but the questionnaire was administered only at the third session. The learning by the teacher education students was not formally evaluated, but informal observations and feedback provided some insights.

As director, this author created overall lesson plans and taught the initial total group lessons. For the first two sessions, we worked as a total group, introducing the first two of seven

games on Moptown Parade. During the remaining sessions, the students let the children progress at their own rates and had the children alternate on- and off-computer activities. This enabled us to sustain interest for an hour as well as try to build particular skills as designed in the teacher's packet of materials accompanying the program. Otherwise, total group explanations were provided only when the most difficult of the seven games "Clubhouse" was introduced and when Teddy's Playground was introduced as a new package.

The first six games on Moptown Parade increase in difficulty and build on one another leading up to the seventh game. The first four are quite easy for a second or third grader and not really difficult for an average or above first grader. These six games are, in order: "Make My Twin, Who is Different? How are We Alike? Make My Opposite, Who Comes Next and Moptown Parade. The Clubhouse game requires combining previously learned skills and deduction to determine the logical rule that allows moppets to enter the clubhouse. This can be played at two levels, with one or two rules. The moppet creatures called Bibbits and Gribbits possess four traits: that of being Bibbits (with a big foot and big nose) or Gribbits (with a tails and say, "grrr"); being either red or blue, tall or short and fat or thin. The games involve identifying and using these traits to make twins, opposites, determine who is different, follow patterns or, by process of elimination in as few steps as possible, determine the trait that allows a creature into the clubhouse. Many of the earlier skills are reading readiness skills or skills of looking for details and the worksheets that accompany the games attempt to help the children transfer these skills to either reading or math. These exercises have them assess which word is different, or which math problem uses subtraction instead of addition, or ask them to continue a pattern of numerals or letters.

Teddy's Playground is a similar package of three increasingly difficult games, using the word "attributes" (instead of "traits") and a playground theme. Hence, there is a swing game, a see-saw game and a slide game. One is a sorting game, another has the children making a change of attribute to put objects in order, and the third uses analogies. Attributes are shape, color and shading. The challenge is to play several games correctly before the reward of seeing the teddy bear swinging, see-sawing or sliding is presented. Even very young children who have more difficulty moving the objects around the screen, seem to have the patience to keep trying until they successfully get a moving teddybear. Teddy's Playground was not introduced until all children had successfully played at least the first six games. Once it was introduced and they had all tried the Clubhouse game, the children were allowed to freely choose between the two programs playing at whatever level they enjoyed. Very simple, immediate mode turtle graphics of Logo were introduced at the last session and the children were free to try it out or continue with the other programs until the time ran out. For rewards on the paperwork, Bibbits and Gribbits as well as other appropriate figures were designed on Printshop (Balsam, 1984) and printed out as stickers for completed papers. The children were introduced to and started the supplementary worksheets during each session, but were encouraged to complete unfinished papers at home and bring them back to the next session. This procedure allowed for maximum computer time, but also supplied sufficient practice time with the skills to ensure some reinforcement and possible transfer of learning.

Project Results

Although the project was generally deemed successful in terms of meeting the goals and in terms of the children's enjoyment, there were some observations and insights gained that bear mentioning. The results of the paper and pencil tests and the questionnaires are summarized below. Since this project was repeated several times, there were changes made each time based on experiences from the previous time. Also, as time allowed, the testing approaches became more sophisticated. Therefore, each offering's results are reported separately and no attempt has been made to combine the figures.

Below are the results of the various tests, post or pre and post for the three times the course was offered. Also included are the results of the questionnaire to the students at the end of the first and third offerings. Shortness of time during the summer did not permit giving the questionnaire during the summer. Although attendance was very good throughout the various offerings, some were missing at the end of the project and so there are fewer test and questionnaire results than there were children in the project.

Figure 1. Pre-Post-test Scores

Offering 1 – Spring 1986				N = 6		Possible score = 16	
Student	Pre	Post	Grade	Pre $\bar{X}$	s.d.	Post $\bar{X}$	s.d.
1		16	2			14.0	1.41
2		15	2				
3		14	2				
4		12	2				
5		14	1				
6		12	1				
Offering 2 – Summer 1986				N = 8		Possible score = 16	
Student	Pre	Post	Grade	Pre $\bar{X}$	s.d.	Post $\bar{X}$	s.d.
1	16	16	3	11.5	3.42	13.0	2.39
2	16	13	3				
3	11	14	3				
4	9	11	2				
5	7	12	2				
6	14	16	1				
7	10	9	1				
8	9	13	1				
Offering 3 – Fall 1986				N = 8		Possible score = 19	
Student	Pre	Post	Grade	Pre $\bar{X}$	s.d.	Post $\bar{X}$	s.d.
1	18	16	3	14.8	3.35	16.88	1.31
2	18	18	3				
3	16	16	3				
4	19	17	2				
5	18	16	2				
6	15	19	2				
7	11	17	2				
8	8	16	1&2				

Wilcoxin Matched Pairs Test showed no significant difference between the pre- and post-test means for either the Summer or Fall group.

Figure 2. Summary of Questionnaire to Children

Fall 1986 N = 6				Spring 1986 N = 8			
Enjoy B & G?	Yes	5	No	1	Yes	8	No 0
Which game enjoyed most?	Moptown Parade	2		Moptown Parade		1	
	Clubhouse	3		Clubhouse		5	
	Teddy Bear	1		Make My Opposite		8	
				Who Comes Next		1	
Why?	"It's fun" "Just liked it" "Challenging"			"Because there was a pattern and one different" "It was hard" "It was fun" "Find one that belongs" "Fun, neat, rules"			
	"Makes music" "Moving marker"						
Enjoy papers?	Yes	4	No	2	Yes	6	No 2
Enjoy Teddy's Playground?	Yes	6	No	0	Yes	8	No 0
Like to play again?	Yes	6	No	0	Yes	8	No 0
Like Computers?	Yes	6	No	0	Yes	8	No 0
Able to boot disk?	Yes	6	No	0	Yes	8	No 0
What would you like to do another time?	Play games			5	Write stories		1
	Write programs			2	Write programs		8
	Play B & G			1	Play B & G		2
	Using computers			3	Play with computers		4
Best thing about the course?	Everything			1	Playing games		1
	Teddy's Playground			1	B & G		2
	Got to do things			1	It was fun		2

Insights gained about the two programs were as follows: 1) the children enjoyed both Moptown Parade and Teddy's Playground. Indications from the questionnaires are that Teddy's Playground was as popular as Moptown Parade and that the most challenging games were generally the most popular, regardless of grade level. 2) It is difficult to keep the children from moving on to the next game in a sequenced set of games such as presented in Moptown Parade. They want to play them all the first session. Since they build on one another, it is imperative to "hold them back" somewhat, but also to individualize as much as possible and allow individuals to progress at their own rate. 3) Since we had three grades represented each time the course was offered, we probably had a greater challenge for individualizing that would a teacher of one grade, though it is fairly obvious from our results that grade was not the sole determining factor in interest or ability to play the games successfully. 4) Because the games are designed to be played successfully at several levels, it is possible for all children to succeed at all games in both packages, but some will succeed better than others because of skills acquired. For example, all could play "Clubhouse" with one rule, but some could use strategies to figure out the rule in fewer tries than others. 5) Playing "Clubhouse" with two rules challenged all but the very brightest of the second or third graders and was probably just too difficult for any first grader. 6) Teddy's Playground goes from sorting, to changing attributes to analogies. Changing attributes at the highest level and the analogies are the most difficult and challenge bright second and third graders. Most of the children could do the sorting game quite successfully after a few experiences with it. 7) Almost all the children chose at some time or another to do the easiest games on the programs rather than accept the challenges of the more difficult ones in a steadily increasing set of challenging experiences. This seems in keeping with this author's experience in teaching this age group non-computer lessons.

Some insights regarding testing of learning achieved (see Figure 1) are as follows: 1) Most of the children who came to these courses were pretty capable children looking for enrichment opportunities; therefore, it was quickly discovered that when they came to the project they had many of the skills being tested for, such as seeing likenesses and differences. 2) The skills not already in place were those of seeing and being able to continue patterns of numerals or letters, changing traits, and using deduction to identify which traits were the rule for inclusion. 3) pre-test scores were a very high mean, so post test scores could not be significantly different. It was

necessary to devise a test using shapes rather than Bibbits and Gribbits to test for deduction skills in the Clubhouse game. Some gains were made on this question between pre- and post-testing, but again no statistically significant differences were found. 4) Some of the observably most capable students made mistakes on the post-test because they couldn't be bothered taking the time to work carefully. Again, this author has seen this phenomenon with other children of this age and ability when teaching non-computer lessons. 5) It is clear from both observation and the questionnaire given the children (see Figure 2) that they enjoy computers, that they like educational games on the computer, that most like to be challenged most of the time, and that at least at this point in time, children are attracted to computer offerings, they can learn new skills and they want to come again and play games, write programs, or write stories on the computer. 6) It does not seem to make any difference how the course is offered, once a week for six or seven weeks, or five consecutive days in one week. Scores from all three groups were similar.

Some insights gained from the university students involved are as follows: 1) they felt it was worthwhile and they learned from the modeling done by the project director as well as their individual experiences with the different children. 2) they felt the lesson plans were helpful in seeing the purposes and procedures for each session. 3) the few who had opportunities to try group lessons to introduce an aspect of the program felt challenged to explain the game adequately and to hold the students' interest as a group. 4) they recognized the vast difference in the children in terms of maturity, skills, and attention span, and realized that even when children choose to be at an activity, they must be challenged, activities must be varied and different children will prefer different activities and respond differently to the varied aspects of the entire program of on- and off-computer activities. 5) all wish to participate next semester in the next project offered.

Conclusions

Both Moptown Parade and Teddy's Playground are viable problem-solving programs that 1st, 2nd and 3rd graders enjoy and seem to benefit from in terms of learning some new skills. Statistically, there were not significant changes over the time these programs were used. Perhaps this is because the children in these groups by and large had many of the skills when they came to the program or perhaps it is because the time period and practice with the skills was too short. It may also be that the tests were inadequate measures of any learning that did take place. It is clear from observation that the children enjoyed and looked forward to their time on the computers. They did not appear bored, nor ask for other things to do. They basically sustained interest as long as there were several games from which to choose within the hour and an opportunity for a change of pace provided by off-computer activities and explanations. If a classroom teacher were to use these programs, it would be important to provide as much individualization as possible and to allow for self-pacing of working through the programs. It would also be important and probably realistic to allow for shorter periods of time than one hour for a given child or group of children to work with the programs. The Bibbits and Gribbits and playground themes of the two programs are appealing to children of these ages and with different challenge levels the programs appear to be suitable for the prescribed grade levels of 1st through 3rd, although this author also has successfully used the easiest levels of both programs with four and five year olds.

References

- Balsam, D. (1984) *Printshop* [Computer program]. San Rafael, CA: Broderbund Software, Inc.
- Grimm, L. (1981) *Moptown Parade* [Computer program]. Menlo Park, CA: The Learning Company/Addison-Wesley Publishing Company.
- Irby, E. G. (1985, January). Temperature Conversions: A Problem-Solving Computer Activity. *Electronic Learning*, p. 48.
- Parker, J. A., Barry, M., & Exner, B. (1984, May/June) Study Says 'Students Prefer Problem-solving CAP'. *Electronic Education*, p. 18.
- Stanger, D. (1985) *Teddy's Playground* [Computer program]. Pleasantville, NY: Sunburst Communications, Inc.